**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**
**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ**
**імені ІГОРЯ СІКОРСЬКОГО»**
**Інститут прикладного системного аналізу**
**Кафедра математичних методів системного аналізу**

До захисту допущено
В. о. завідувача кафедри
_____ О.Л. Тимощук
«___»_____2020 р.

# Дипломна робота

**на здобуття ступеня бакалавра**
**за освітньо-професійною програмою «Системний аналіз і управління»**
**спеціальності 124 «Системний аналіз»**
**на тему: «Статистична модель для визначення ймовірності відхилення рекурентних платежів»**

Виконав:
студент IV курсу, групи КА-63
Мілантьєв Сергій Сергійович _____

Керівник:
асистент
Макуха Михайло Павлович _____

Консультант з економічного розділу:
доцент, к.е.н., доцент кафедри ТТПЕ
Шевчук Олена Анатоліївна _____

Консультант з нормоконтролю:
доцент, к.т.н., доцент кафедри ММСА
Коваленко Анатолій Єпіфанович _____

Рецензент:
доцент, к.т.н., доцент кафедри СП
Харченко Константин Васильович _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2020 року

**Національний технічний університет України**
**«Київський політехнічний інститут імені Ігоря Сікорського»**
**Інститут прикладного системного аналізу**
**Кафедра математичних методів системного аналізу**

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 124 «Системний аналіз»

Освітньо-професійна програма «Системний аналіз і управління»

ЗАТВЕРДЖУЮ
В.о.завідувача кафедри
_____ Оксана ТИМОЩУК
«___»_____20__ р.

**ЗАВДАННЯ**
**на дипломну роботу студенту**
**Мілантьєву Сергію Сергійовичу**

1. Тема роботи «Статистична модель для визначення ймовірності відхилення рекурентних платежів», керівник роботи Макуха Михайло Павлович, затверджені наказом по університету від «25» травня 2020 р. № 1143-с

2. Термін подання студентом роботи 08 травня 2020 року _____

3. Вихідні дані до роботи: відкритий набір даних 900 000 платежів

4. Зміст роботи: огляд предметної області, розгляд практичних застосувань систем підрахунку ймовірностей, обґрунтування застосування методів машинного навчання, практичні результати, отримані на вихідних даних, та їх аналіз.

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо)

6. Консультанти розділів роботи*

| Розділ | Прізвище, ініціали та посада консультанта | Підпис, дата | |
|---|---|---|---|
| | | завдання видав | завдання прийняв |
| Економічний | Шевчук О. А., доцент | | |

7. Дата видачі завдання _____

Календарний план

| № з/п | Назва етапів виконання дипломної роботи | Термін виконання етапів роботи | Примітка |
|---|---|---|---|
| 1 | Формування теми дослідження | 21.01.20 – 18.02.20 | Виконано |
| 2 | Аналіз існуючих досліджень на дану тему | 18.02.20 – 15.03.20 | Виконано |
| 3 | Формування задачі дослідження | 15.03.20 – 24.03.20 | Виконано |
| 4 | Узгодження теми | 24.03.20 | Виконано |
| 5 | Пошук та збір даних | 24.03.20 – 08.04.20 | Виконано |
| 6 | Розробка програмного продукту | 08.04.20 – 14.05.20 | Виконано |
| 7 | Аналіз результатів | 14.05.20 – 17.05.20 | Виконано |
| 8 | Оформлення пояснювальної записки | 17.05.20 – 27.05.20 | Виконано |
| 9 | Підготовка презентації | 27.05.20 – 01.06.20 | Виконано |
| 10 | Попередній захист дипломної роботи | 01.06.20 – 03.06.20 | Виконано |
| 11 | Захист дипломної роботи | 16.06.20 | |

Студент Сергій Сергійович Мілантьєв

Керівник Михайло Павлович Макуха

* Якщо визначені консультанти. Консультантом не може бути зазначено керівника дипломної роботи.

# РЕФЕРАТ

Дипломна робота: 93 с., 15 рис., 8 табл., 2 додатки, 6 джерел.

МАШИННЕ НАВЧАННЯ, БІНАРНА КЛАСИФІКАЦІЯ, ПРОГНОЗУВАННЯ.

Об’єкт дослідження – набір даних, який містить 900 000 записів про користувачів та їх рекурентні платежі.

Предмет дослідження – методи машинного навчання та їх оптимізація за допомогою підбору гіперпараметрів.

Мета роботи – розробити якісну модель для визначення вірогідності відхилення рекурентного платежу, яка буде задовольняти всім умовам для високоякісної роботи, а саме бути точною, достатньо відказостійкою та швидкою.

Актуальність – оптимізування маркетингових кампаній та вдосконалення системи прийняття рішень в компанії в цілому.

Шляхи подальшого розвитку предмету дослідження – прогнозування рекурентних платежів, які ще не є заплановані проте можуть відбутися у майбутньому.

# ABSTRACT

Diploma work: 93 p., 19 fig., 9 tabl., 2 appendixes, 24 sources.

STATISTICAL MODEL TO DETERMINE THE PROBABILITY OF RECURRING PAYMENTS REJECTION

The object of the study – data set that contains 900,000 records about users and their recurring payments information.

The subject of research – methods of machine learning and their optimization through the selection of hyperparameters.

The purpose of the work – develop a precise model to determine the probability of recurring payments rejection, which will meet all the conditions for high quality work, accurate, sufficiently fault tolerant and fast.

Actuality – optimizing marketing campaigns and improving the decision making system in the company as a whole.

Ways of further development of the subject of research – forecasting recurrent payments, which are not yet planned but may occur in the future.

ЗМІСТ

# ВСТУП

Разом з розвитком інформаційних технологій та створенням великої кількості стартапів змінюються і бізнес моделі. В сучасному світі є безліч різних моделей монетизації бізнесу: прямі продажі, підписка, реклама тощо. На сьогоднішній день дуже популярною є модель платної підписки: користувачі мають змогу отримувати цифрову послугу протягом того терміну за який вони заплатили, після закінчення цього періоду проходить автоматичний рекурентний платіж, що дає змогу продовжувати користуватися сервісом.

Достеменно не відомо хто є автором першої підписної моделі, але є данні котрі свідчать, що ще у 17 столітті видавці книг використовували її. З часом ця бізнес модель стала основною при наданні послуг кабельного телебачення, щоденних газет та іншого. Але масово цією моделлю стали користуватися разом з розвитком інтернет сервісів, користувачі яких не хотіли відразу платити великі кошти, але були згодні раз в якийсь час сплачувати невелику суму грошей. Компанії це також влаштовувало, адже вони отримували більшу кількість користувачів їх сервісом і мали розуміння скільки орієнтовно коштів вони отримають в майбутньому.

Для ефективного планування та прийняття правильних рішень в компаніях часто використовують різні методи та підходи для прогнозування майбутніх надходжень грошей. Розуміння того, який буде прибуток наступного тижня, місяця, кварталу можна більш зважено підходити до питань розширення бізнесу або його оптимізації, оцінювати ризики інвестицій у нові проекти та аналізувати існуючі.

При використанні підписної моделі розрахунок вірогідного прибутку зводиться до прогнозування відхилення вже існуючих рекурентних платежів та надходжень від нових клієнтів. Прибуток від нових клієнтів дуже щільно

залежить від багатьох факторів таких як маркетингові компанії, лояльність бренду у людей, позиціювання у ЗМІ та багато іншого, тому в даній роботі я сконцентруюся на першій складовій. Адже маючи вірогідності для кожного платежу та їх величини можна отримати орієнтовну суму надходжень коштів від існуючих клієнтів у майбутньому.

Метою роботи є дослідження різних методів та підходів для визначення ймовірностей відхилення рекурентних платежів на основі існуючих даних о користувачеві.

# РОЗДІЛ 1 ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ

## 1.1 Основні поняття

Перш за все визначимось з поняттям бізнес модель та монетизація.

Бізнес-модель – це план компанії щодо отримання прибутку. Він визначає продукцію чи послуги, які продаватиме бізнес, цільовий ринок, який він визначив, та витрати, які він передбачає.

Новий бізнес, що розвивається, повинен мати модель бізнесу, хоч лише для того, щоб залучити інвестиції, допомогти йому набрати таланти та мотивувати персонал. Створеним бізнесам доводиться часто переглядати та оновлювати свої бізнес-плани, інакше вони не зможуть адаптуватися під нові тенденції та проблеми, що стоять перед ним. Інвесторам необхідно переглянути та оцінити бізнес-плани компаній, які їх цікавлять.

Бізнес-модель – це план високого рівня для вигідної діяльності певного бізнесу на певному ринку. Основною складовою бізнес-моделі є ціннісна пропозиція. Це опис товарів або послуг, які компанія пропонує, і чому вони зацікавлять клієнтів, чим відрізняється товар або послугу від своїх конкурентів.[1]

Бізнес-модель для нового підприємства повинна також охоплювати прогнозовані витрати на запуск та джерела фінансування, цільову базу клієнтів для бізнесу, маркетингову стратегію, огляд конкуренції та прогнози доходів і витрат.

Монетизація означає зробити прибуток вигідним. Цей термін став досить популярним в останні роки і часто відноситься до цифрових активів. Такий вміст може включати програмне забезпечення, цифрові носії інформації або файли. З поширенням інтернет-медіа та цифрових активів це стало

популярною концепцією серед власників веб-сайтів, розробників програмного забезпечення та інших постачальників цифрового контенту.

Поширені методи монетизації включають стягнення плати за кінцевий продукт або використання реклами та афілійованого маркетингу для отримання доходу за безкоштовні послуги чи продукти. Наприклад, власник веб-сайту може створювати медійні реклами для монетизації веб-сайту, постачальник вмісту зі списком електронних листів може містити партнерські пропозиції щодо монетизації своїх послуг, а розробник програмного забезпечення може постачати свій продукт з іншими продуктами, щоб отримати частку доходу.

### 1.1.1 Огляд підписної моделі бізнесу

Підписна бізнес-модель – це бізнес-модель, в якій замовник повинен регулярно платити ціну, що повторюється, за доступ до товару. Ця модель була започаткована видавцями книг та періодичних видань у 17 столітті, і зараз її використовують багато підприємств та веб-сайти.

Замість того, щоб продавати товари окремо, підписка пропонує періодичне (щомісячне, щорічне або сезонне) використання або доступ до товару чи послуги, або, у випадку організацій, орієнтованих на ефективність, таких як оперні компанії, квитки на весь набір певного набору кількість (наприклад, п'ять-п’ятнадцять) запланованих виступів на весь сезон. Таким чином, разовий продаж товару може стати повторюваним і може розвинути лояльність до бренда.

Галузі, які використовують цю модель, включають сервіси з продажу музики, приватні провайдери веб-пошти, кабельне телебачення, супутникове телебачення з платними телевізійними каналами, супутникове радіо,

телефонні компанії, оператори мобільної мережі, Інтернет-провайдери, видавці програмного забезпечення, веб-сайти (наприклад, веб-сайти, що ведуть блоги), постачальників бізнес-рішень, фірм, що надають фінансові послуги, клубів охорони здоров'я, послуг із скошування газону та снігоочищення та фармацевтичних препаратів, а також традиційних газет, журналів та наукових журналів.

Поновлення підписки може бути періодичним і активуватися автоматично, так що вартість нового періоду автоматично оплачується заздалегідь авторизованою платою на кредитну карту або на чековий рахунок. Поширеною варіацією моделі в онлайн-іграх та на веб-сайтах є модель freemium, в якій перший рівень вмісту є безкоштовним, але доступ до преміальних функцій (наприклад, ігрових бонусів або архівів статей) обмежений платними абонентами.

Freemium – це тип бізнес-моделі, що передбачає можливість клієнтам користуватися базовими, так і додатковими послугами. Компанія надає прості та основні послуги користувачеві безкоштовно. За додаткову оплату компанія надає доступ до особливих функцій та сервісів.

Термін freemium приписується Джаріду Лукіну з Алакра, постачальника корпоративних відомостей та інструментів робочого процесу, який створив його у 2006 році. Практика, однак, починається з 1980-х.

За моделлю freemium бізнес надає споживачеві послуги безкоштовно, як спосіб встановити фундамент майбутніх угод. Пропонуючи послуги базового рівня безкоштовно, компанії вибудовують стосунки з клієнтами, в кінцевому підсумку пропонуючи їм розширені послуги, доповнення, розширені ліміти зберігання чи використання.[2]

### 1.1.2 Процес здійснення платежів

Онлайн-платіж – можливість оплатити товари або послуги в інтернеті або через мобільний додаток без використання банкнот. Зазвичай здійснюється прямо на сайті або в мобільному додатку продавця.

Як правило, алгоритм онлайн-платежу виглядає так:

- покупець вибирає товар або послугу;
- продавець підсумовує покупки і виставляє рахунок;
- покупець погоджується сплатити рахунок і підтверджує це введенням банківських чи інших платіжних реквізитів;
- банк або платіжний провайдер підтверджує особу покупця і наявність коштів на рахунку для оплати покупки;
- продавець підтверджує покупку, а банк або платіжний провайдер передає грошові кошти на рахунок продавця.

## 1.2 Рекурентні платежі

Рекурентні платежі – це особливий вид платежів які регулярно виконують списання коштів з банківської картки користувача сервісу без необхідності кожного разу вводити заново реквізитів картки і без особистої участі платника для здійснення операції.

Рекурентні платежі були введені системами VISA / MasterCard і мають відмінні показники безпеки. Користувачеві досить лише раз налаштувати розклад оплат (створити «батьківський» платіж), після чого наступні списання будуть відбуватися автоматично.

### 1.2.1 Зручність рекурентних автоматичних платежів

Даний вид платежів надає можливість компаніям списувати кошти з банківської картки клієнта або електронного гаманця без додаткового підтвердження кожної транзакції. Дана послуга дуже зручна при таких регулярних щомісячних оплатах, як:

- продовження доменних або хостингових послуг;
- підписка на сервіси прослуховування музики, подкастів, перегляду фільмів;
- оплата доступу в інтернет;
- підписка на новинний контент;
- оплати ЖКГ послуг;
- поповнення гаманця ігровий валютою або продовження підписки на платний ТВ контент, при зниженні мінімального залишку на рахунку клієнта;
- автоплатіж часто використовують таксопарки, для списання коштів як з водіїв (оренда автомобіля), так і з пасажирів після надання послуги.

Для клієнта автоплатіж – це позбавлення від головного болю з приводу своєчасної оплати послуг. Для бізнесу – спосіб підвищити прибуток і утримати покупця.

### 1.2.2 Недоліки рекурентних платежів

Компанії люблять модель підписок не тільки через прогнозований прибуток і лояльних клієнтів. У більшості випадків підписки влаштовані таким чином, що від них непросто відмовитися.

У випадку з контентом користувачам здається, що можливості безмежні – можна послухати всю музику і подивитися всі серіали.

Насправді ж багато хто дивиться лише найбільш значущі і відомі серіали, які вже популярні, а не ексклюзивні проекти сервісів. У той же час статистика говорить про те, що люди витрачають на стрімінг-сервіси все більше і більше часу.

Але що станеться, якщо на кілька місяців зупинити підписку на Apple Music або iCloud? Зібрані роками вручну плейлисти та збережені в хмарі файли зникнуть.

Завдяки масивній бібліотеці сервіси типу Netflix створюють ілюзію, що все доступно завжди і в будь-який момент. Але таким же чином через пару десятиліть сервіс може закритися в один день і забрати цю бібліотеку з собою, залишивши клієнтів ні з чим.

Інший недолік – в ліцензованому контенті, який може пропасти ще до того, як користувач встигне його додивитися. Багато хто купляють підписку в сервісах лише заради конкретних відомих серіалів, але стороннє шоу може в будь-який момент зникнути з сервісу через закінчення прав або за бажанням правовласника.

Це сталося з «Офісом» і «Парками та зонами відпочинку», які Netflix не зміг продовжити, тому що власник забрав серіали на власний сервіс відеострімнгу. Те ж саме чекає навесні 2020 року і «Друзів» – їх заберуть в HBO Max.

Купуючи фільм на диску, завжди знаєш, що отримаєш тут і зараз, і за яку ціну. А підписка на сервіс – красива, зручна і дешевша, але лише ілюзія контролю. Якщо в першому випадку покупець платить один раз за конкретний товар, то в другому він віддає гроші за вхід в магазин з безліччю безкоштовних продуктів – ними можна користуватися, але вони залишаться на полицях.

Підписка на онлайн-кінотеатри відкриває доступ до величезної кількості легального контенту, але найчастіше користувачі платять лише за обіцянку компанії регулярно випускати щось нове і якісне. Цього може і не відбутися.

Насправді величезна бібліотека і постійне підгодовування клієнтів новинками – експлуатація первісних інстинктів людини. Разом ці чинники створюють психологічний ефект упущеної вигоди, який не дозволяє так просто змусити себе відписатися: починаєш відчувати, що багато втратиш.

Ефект експлуатують навіть в товарних підписках. Наприклад, якщо клієнт платить Costco за платне членство, то він відчуває себе зобов'язаним відвідати магазин, щоб обґрунтувати собі ці витрати. А вже в офлайні знайде вигідні пропозиції спеціально для передплатників мережі.

У випадку з звичайною покупкою вартість продукту вже не зміниться, адже він вже придбаний. Підписки ж можуть вирости в ціні в будь–який момент: наприклад, Netflix з початку своєї роботи вдвічі підняв ціну на тарифи.

Приклад Netflix показав, що зростання цін не провокує користувачів на масові відписки, а лише впливає на залучення нових клієнтів – люди часто не помічають невеликі платежі, поки не починають складати їх за рік. Зараз багато компаній працюють в мінус, але як тільки вони наберуть критичну кількість клієнтів, ціни виростуть.

### 1.2.3 Причини відхилень рекурентних платежів

Компанія, котра має підписну модель бізнесу, не може гарантувати собі, що кожний клієнт який має підписку на теперішній час, продовжить її, або його рекурентний платіж буде вдалий. Основними причинами не проходження запланованих платежів від користувача є:

- користувач сам відписався від підписки;
- на картці користувача немає коштів;
- картка користувача заблокована;
- картка користувача викрадена;
- платіжна система відхилила платіж;
- закінчився термін придатності платіжної картки.

## 1.3 Оптимізація маркетингових компаній

З часів створення перших маркетингових промо-кампаній, працівники в цій сфері завжди мріяли точно оцінити вплив кожної окремої реклами на користувачів та який прибуток принесе кожна з них. В більшості випадків рекламні кампанії оцінюються через великий проміжок часу після її закінчення, що є дуже не ефективно, адже не можливо приймати рішення в процесі, коли є ще можливість внести вдосконалення. Відсутність можливості ефективно корегувати рекламні компанії призводить до того, що велика їх частка виявляється не ефективними.

З приходом цифрових видів реклами, таких як, реклама в фейсбуці, гугл, инстаграммі та на інших цифрових площадках в мережі інтернет стало можливо більш глибоко оцінювати показники ефективності реклами. Зараз усі

площадки надають досить детальну та зрозумілу статистику кожної рекламної компанії, що дає змогу швидко та ефективно аналізувати та приймати рішення.

На перших етапах розвитку молодої компанії для аналізу маркетингових компаній достатньо вбудованих можливостей площадки, що надає рекламні послуги. Рекламний кабінет фейсбуку вважається одним з найбільших по можливостям для аналізу рекламних кампаній, та можливостям налаштування її параметрів. Існують навіть спеціальні курси для освоєння всіх його можливостей.

Проте з ростом компанії з'являється необхідність більш глибоко оцінювати маркетингові кампанії, наприклад, прогнозуючи скільки коштів принесе та чи інша реклама. Це стає можливо оцінюючи скільки принесе коштів у майбутньому кожний окремий користувач, групуючи потім їх по каналам трафіку звідки вони дізналися про послугу компанії.

Маючи інформацію про те, чи принесе достатньо коштів кожна з кампаній, можна приймати рішення про оптимізацію кампаній, які показують низькі результати, та масштабувати – високоефективні, для зменшення витрати та збільшення прибутку.

При наявності у компанії підписної моделі бізнесу досить зрозуміло як звідки та коли можуть прийти кошти – рекурентні платежі. Навчившись достатньо добре прогнозувати їх відхилення, можна отримувати гарні прогнозні розрахунки прибутку компанії на наступний період часу.

## 1.4 Висновки до розділу 1

У цьому розділі ми ознайомилися з основними поняттями про бізнес моделі та платежі, а також була розглянута задача, яку буде розглянуто у наступних розділах. Задача складається з визначення ймовірності відхилення рекурентних платежів для подальшого аналізу та прийняття рішень стосовно маркетингових кампаній.

У наступних розділах ми розглянемо різні підходи у розробці статистичної моделі та оберемо найкращу модель для вирішення поставленої задачі.

# РОЗДІЛ 2 МАТЕМАТИЧНЕ ОБҐРУНТУВАННЯ ВИКОРИСТОВУВАНИХ МЕТОДІВ МАШИННОГО НАВЧАННЯ

## 2.1 Вступ до розділу 2

З появою великої кількості відкритих великих баз данних та дешевих, але водночас достатньо потужних обчилювальних пристроїв, швидкими темпами стали розвиватися методи машинного навчання.

Машинне навчання (МН) – це навчання комп'ютерної програми або алгоритму поступового поліпшення виконання поставленого завдання. З дослідницької боку машинне навчання можна розглядати через призму теоретичного і математичного моделювання процесу його роботи. Проте, на практиці, це вивчення того, як створювати алгоритми, які демонструють ітеративное поліпшення. Типи машинного навчання (Рисунок 2.1) можна виділяти за різними критеріями, але ось основні три:

- Навчання з вчителем – маючи дані у вигляді прикладів з мітками, ми можемо подавати алгоритму їх один за іншим, ортимувати прогнозну мітку і давати зворотний зв'язок: передбачив він правильно чи ні. З часов алгоритм навчиться наближуватися до точного прогнозу взаємозалежностей між прикладами і їх мітками. Будучи повністю навченим, алгоритм зможе передбачити вірну мітку для приклада, який він ніколи раніше не зустрічав.
- Навчання без вчителя – в данному випадку дані не мають позначок. Замість цього алгоритм отримує в своє розпорядження багато, дуже багато даних та інструментів для розуміння їх властивостей. Завдяки цьому він може навчитися групувати і організовувати старі дані в нові.

– Навчання з підкріпленням – термін, який відноситься до групи методів автоматичного самонавчання. Ці методи відрізняються здатністю функціонувати без необхідності в наближеному вирішені проблеми. Навчання проходить як послідовність пробних дій, які поступово призводять до перевірки правильних дій та уникнення невідповідних. Результатом навчання є оптимальна стратегія дій у будь-якій ситуації. Стратегія є оптимальною, якщо їй вдається максимально збільшити суму всіх винагород, отриманих в ході її реалізації.

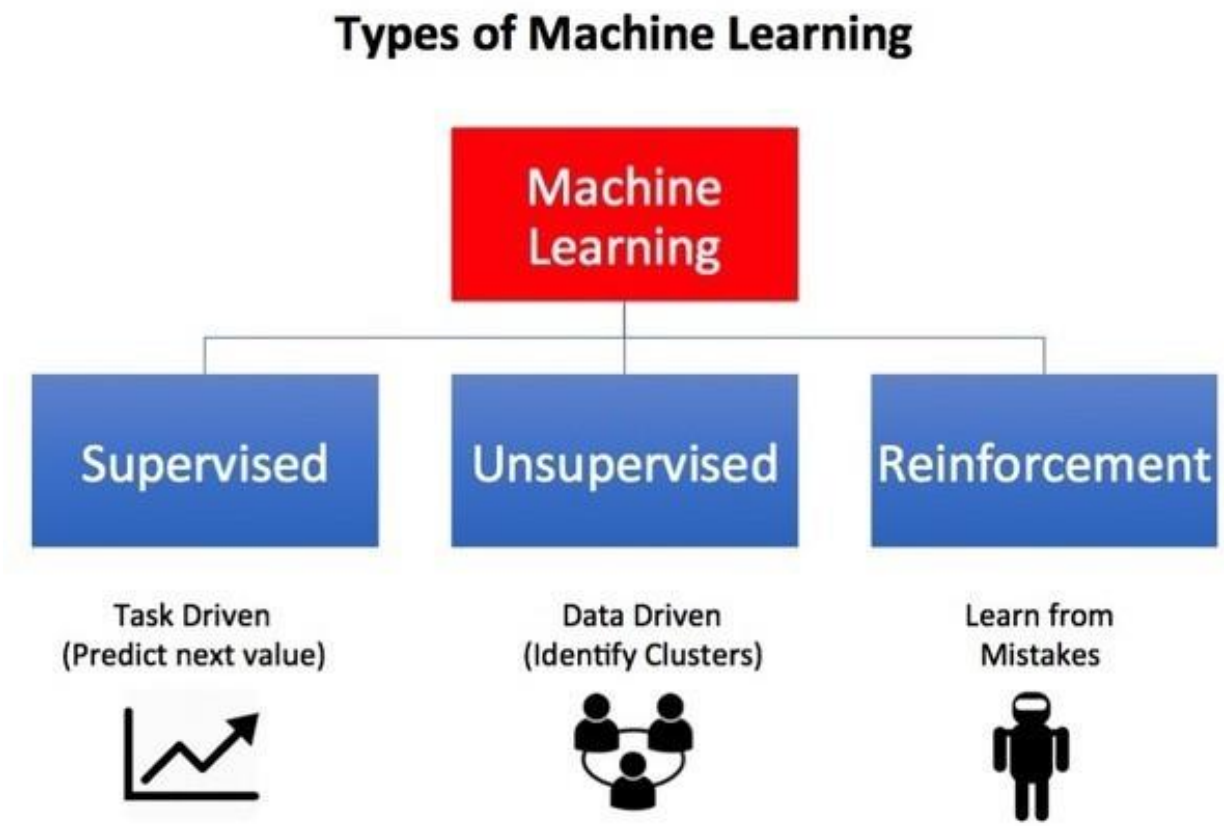



Рисунок 2.1 – Типи машинного навчання [3]

У цьому розділі ми розглянемо різні методи визначення ймовірностей, зокрема дерева рішень, випадковий ліс, логістична регресія, а також різні способи оптимізації їх параметрів.

Отже для навчання всіх моделей у даній роботі ми використовували набір даних, де достеменно відомо чи був відхилений рекурентний платіж.

## 2.2 Лінійна регресія

Лінійна регресія це метод апроксимації залежностей між вхідними та вихідними змінними на основі лінійної моделі. Є частиною більш широкої статистичної методики, званої регресійний аналізом.

Якщо розглядається залежність між одними вхідними і однією вихідною змінною, то має місце проста лінійна регресія. Для цього визначається рівняння регресії y=ax+b і будується відповідна пряма, відома як лінія регресії (Рисунок 2.2).

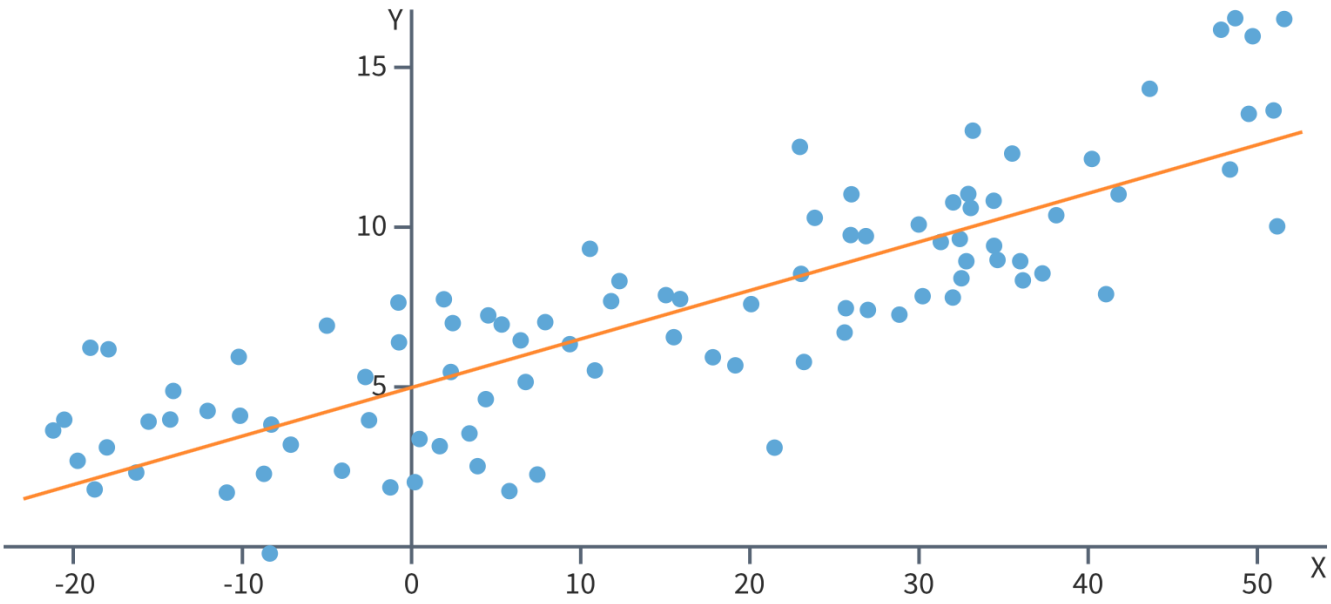



Рисунок 2.2 – Лінійна регресія

Коефіцієнти a і b, так звані – параметри моделі, визначаються таким чином, щоб сума квадратів відхилень точок, що відповідають реальним спостереженням даних, від лінії регресії, була б мінімальною. Коефіцієнти зазвичай оцінюються методом найменших квадратів.

Якщо шукається залежність між декількома вхідними і однією вихідний змінними, то має місце множинна лінійна регресія. Відповідне рівняння виглядає як:

$$Y = b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_n x_n,$$

де n – число вхідних змінних.

Очевидно, що в даному випадку модель буде описуватися не прямою, а гіперплощиною. Коефіцієнти рівняння множинної лінійної регресії підбираються так, щоб мінімізувати суму квадратів відхилення реальних точок даних від цієї гіперплощини.

Лінійна регресія була першим видом регресійного аналізу, який був ретельно вивчений і почав широко використовуватися в практичних додатках. Це пов'язано з тим, що в лінійних моделях оцінювання параметрів простіше, а також з тим, що статистичні властивості отриманих оцінок легше визначити.

Лінійна регресія має багато практичних застосувань. Більшість додатків потрапляють в одну з двох широких категорій:

- Якщо метою є прогнозування, лінійну регресію можна використовувати для підгонки моделі до спостережуваного набору даних;
- Якщо мета полягає в тому, щоб пояснити мінливість вихідної змінної, можна застосувати лінійний регресійний аналіз для кількісної оцінки сили взаємозв'язку між вихідний і вхідними змінними.

### 2.2.1 Регуляризація

Регуляризація – це техніка, яка вносить незначні зміни до алгоритму навчання таким чином, що модель узагальнюється краще. Це, в свою чергу, покращує ефективність моделі і для даних, на яких алгоритм не навчався.

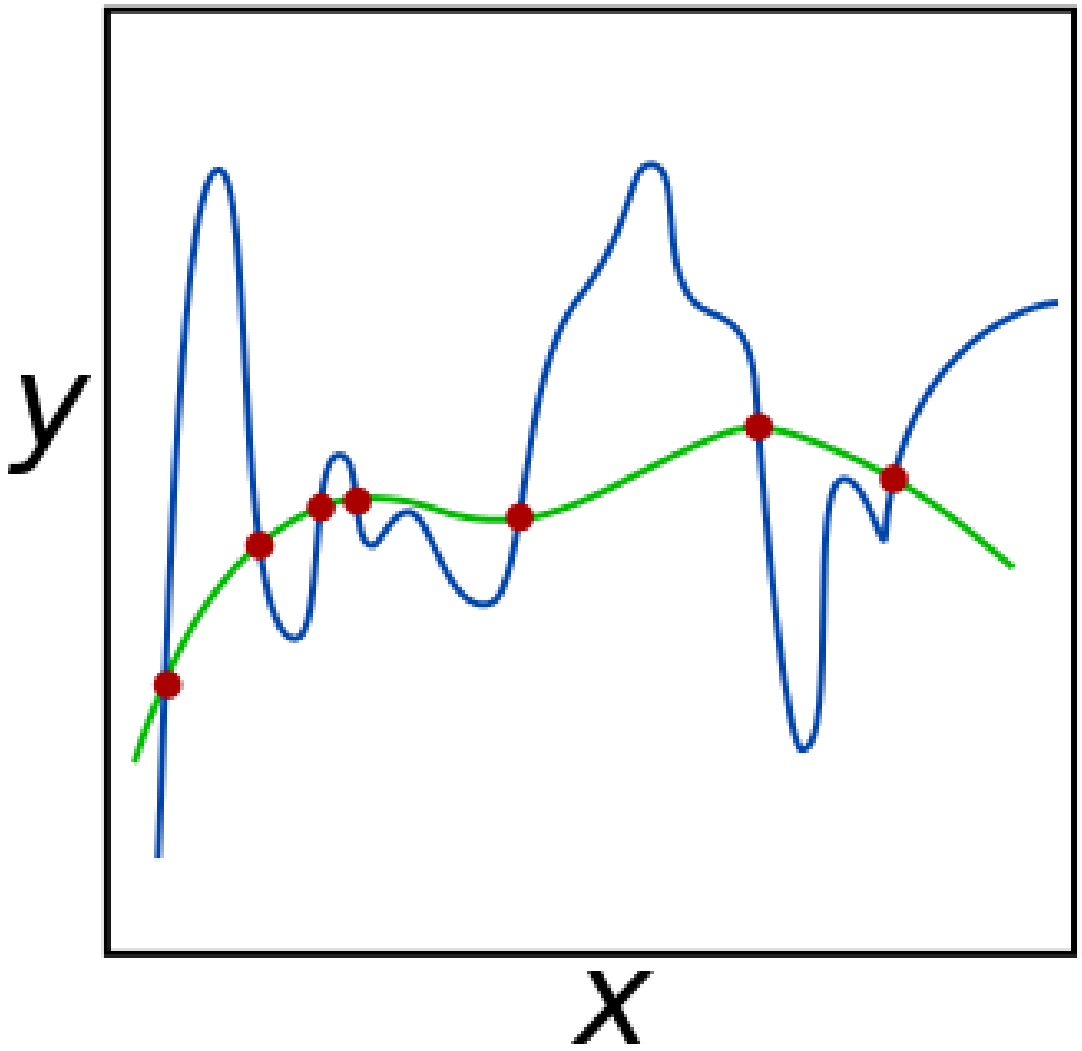



Рисунок 2.3 – Перенавчання

З баєсівської точки зору багато методів регуляризації відповідають додаванню деяких апріорних розподілів на параметри моделі.

Деякі види регуляризації:

– $L_1$-регуляризація:

$$L_1 = \sum_i \left(y_i - y(t_i)\right)^2 + \lambda \sum_i |a_i|$$

– $L_2$-регуляризація:

$$L_2 = \sum_i \left(y_i - y(t_i)\right)^2 + \lambda \sum_i a_i^2$$

Перенавчання в більшості випадків виявляється в тому, що в многочленах утворюються занадто великі значення коефіцієнтів. Відповідно, необхідно додати в цільову функцію штраф за це.

Немає рішення щодо багатокритеріальної оптимізації або оптимізації, в якій область значення цільової функції є простір, на якому немає лінійного порядку, або його важко ввести. Майже завжди знайдуться точки в області визначення функції яку оптимізують і які задовольняють обмеженням, але значення в точках не порівнянні між собою. Щоб знайти всі точки на кривій Парето, використовують скаляризації. В оптимізації регуляризація – це загальний метод скаляризації для завдання двухкрітеріальної оптимізації.

## 2.2 Логістична регресія

Логістична регресія – це статистична модель, яка використовуючи логістичні функції розраховує ймовірності приналежності до кожного класу.

Регресійна модель бінарного вибору – це регресійна модель, в якій залежна змінна дихотомічна (бінарна). Залежна змінна може приймати лише

два значення і означати, наприклад, приналежність до певної групи (надійний клієнт або ненадійний клієнт банку), що роблять дію (покупка товару), варіанти відповіді «так» або «ні» (подобається реклама або не подобається).

Будувати звичайну лінійну регресійну модель з бінарними залежними змінними не можна. У цьому випадку неможливо буде інтерпретувати передбачені по регресії в безперервної кількісної шкалою значення залежної змінної.

Значення факторів в моделях бінарного вибору повинні бути виміряні в кількісній шкалі. Також в моделі бінарного вибору можна включати в якості факторів категоріальні змінні.

Отже, в моделях бінарного вибору будується регресійна модель залежності ймовірності того, що результативна дихотомічна змінна прийме значення 0 або 1 при заданому значенні факторів.

Для моделювання ймовірності дихотомічної залежності змінної підбирають спеціальну монотонно зростаючу функцію, значення якої лежить в проміжку від 0 до 1.

В якості спеціальної функції в моделях бінарного вибору зазвичай використовують:

- логістичну функцію;
- функцію стандартного нормального розподілу.

Моделі бінарного вибору на основі логістичної функції називаються логістичної регресією або логіт-моделлю.

За допомогою логістичної регресії прогнозується ймовірність відгуку для залежної змінної від включених в модель незалежних змінних. На основі прогнозних значень ймовірності можна зробити класифікацію всіх спостережень на дві групи. Окремим аналізом при побудові моделі логістичної регресії є аналіз ROC-кривих (Receiver Operator Characteristic). ROC-аналіз дозволяє вибрати оптимальне значення порогового значення ймовірності для

класифікації. ROC-крива – крива, яка використовується для представлення результатів бінарної класифікації та оцінки ефективності класифікації.

У прикладному статистичному аналізі логістична регресія використовується для вирішення двох завдань: моделювання взаємозв'язку і класифікації спостережень. Логістичну регресію застосовують при проведення клінічних досліджень в медицині, в банківському скоринг для побудови рейтингу позичальників і управління кредитними ризиками, в споживчому скоринг для моделювання поведінки покупців та інших сферах.

## 2.3 Дерево рішень

Дерево прийняття рішень (рисунок 2.4) – це алгоритм з деревоподібною структурою, де кожен внутрішній вузол містить правила для визначення листка, а кожне листя (кінцевий вузол) містить мітку класу.

В результаті перевірки, приклади, які опинилися в одному вузлі, розгалужуються на дві підмножини, в одне з яких потрапляють приклади, що задовольняють правилу, а в інше – що не задовольняють.[4]

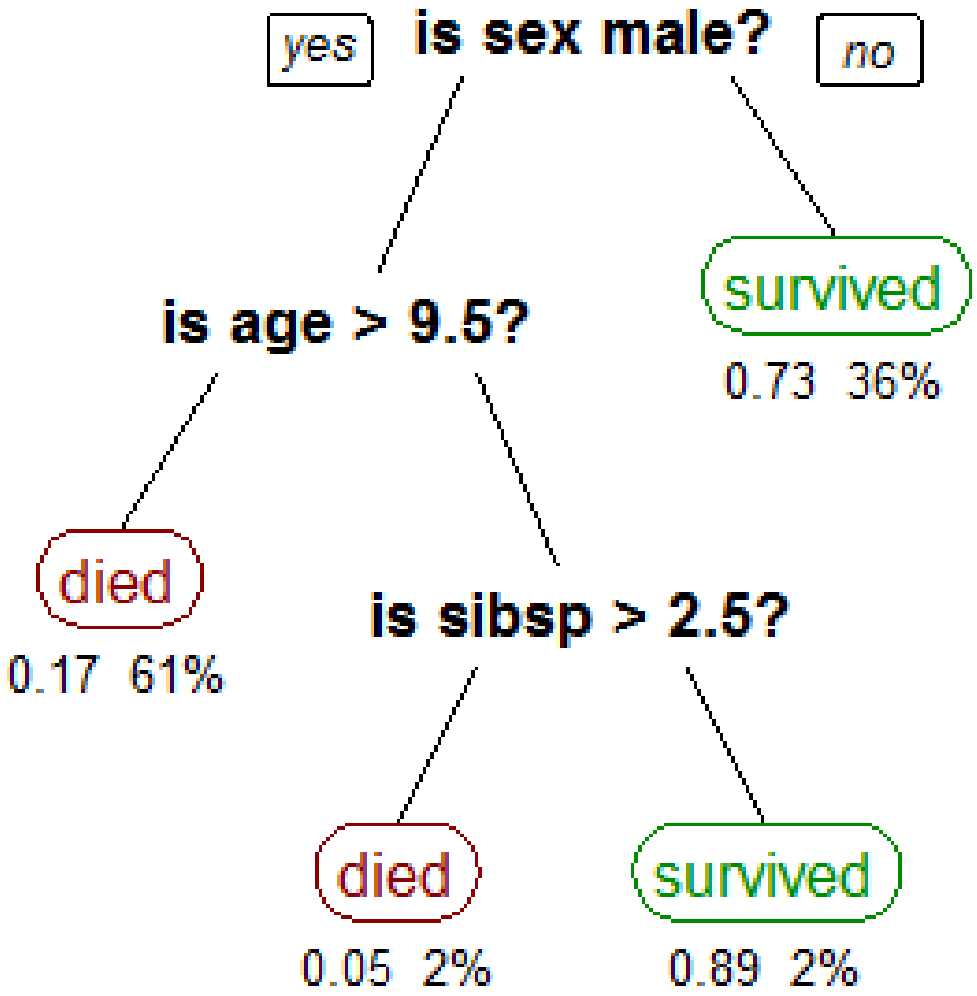



Рисунок 2.4 – Дерево прийняття рішень[5]

Потім рекурсивно до кожної підмножини використовується дана процедура. Це триває доки не будуть виповнені умови зупинки. На кінцевому етапі дії алгоритму розгалуження не відбувається і утворюється лист. Він визначає, яке буду передбачення для всіх прикладів, що попадуть у нього. В задачу регресії – це середнє значення прикладів в листу, в класифікації – мажоритарний клас.

Отже, в листі міститься підмножина прикладів, які підпадають під правила гілки, а вузол це правило виконуючи які, об’єкти потрапляють у листки.

## 2.4 Випадковий ліс

Випадковий ліс – один з найбільш приголомшливих алгоритмів машинного навчання, придумані Лео Брейманом і Адель Катлер. Цей алгоритм дуже популярний, це обумовлено тим, що він дуже універсальний. По перше, він показує гарні результаті при вирішені багатьох проблем, по-друге, його використовують при розгляді таких завдань, як кластеризація, регресія, пошук аномалій, класифікація і т.д.

RF (random forest) – це безліч вирішальних дерев. У задачі регресії їх відповіді усереднюються, в завданню класифікації приймається рішення голосуванням за більшістю. Щоб побудувати випадковий ліс з N вирішальних дерев, необхідно:

1) побудувати за допомогою бутстрапа N випадкових підвибірок;
2) кожна отримана підвибірка використовується як навчальна вибірка для побудови відповідного вирішального дерева. Дерево будується, поки в кожному листку виявиться не більше заданої кількості об'єктів. Дуже часто дерева будують до кінця, тобто залишається один об’єкт, щоб отримати складні і перенавчання вирішальні дерева з низьким зсувом;
3) побудовані дерева об'єднуються в композицію: у завданнях регресії кінцевий результат це середнє значення передбачень усіх дерев, класифікації – мажоритарний клас.

Отже, випадковий ліс – композиція глибоких дерев, які будуються незалежно один від одного. Але такий підхід має наступну проблему.

Навчання глибоких дерев вимагає дуже багато обчислювальних ресурсів, особливо в разі великої вибірки або великого числа ознак.

Якщо обмежити глибину вирішальних дерев у випадковому ліс, то вони вже не зможуть вловлювати складні закономірності в даних. Це призведе до того, що зсув буде занадто великим.

Друга проблема з випадковим лісом полягає в тому, що процес побудови дерев є ненаправленим: кожне наступне дерево в композиції ніяк не залежить від попередніх. Через це для вирішення складних завдань необхідна величезна кількість дерев.

## 2.5 Градієнтний бустинг

Градієнтний бустинг є одним з кращих способів спрямованої побудови композиції. У градієнтному бустингу побудована композиція є сумою, а не їх середнім значенням базових алгоритмів. Це пов'язано з тим, що алгоритми навчаються послідовно і кожен наступний виправляє помилки попередніх.

Ансамбль – це набір моделей, які разом дають відповідь (наприклад, середнє по всім). Є декілька техніки ансамблювання – Беггінг і Бустинг.

Беггінг – навчання базових алгоритмів відбувається на випадкових підвибірках навчальної вибірки. Причому чим менше розмір випадкової підвибірки, тим більш незалежними виходять базові алгоритми.

Бустинг – це підхід до побудови композицій, в рамках якого:

- базові алгоритми будуються послідовно, один за іншим;
- кожен наступний алгоритм будується таким чином, щоб виправляти помилки вже побудованої композиції.

Завдяки тому, що побудова композицій в бустингу є спрямованим, досить використовувати прості базові алгоритми, наприклад неглибокі дерева.

В задачах машинного навчання з учителем головною метою є зменшення значення функції втрат.

Розглянемо принцип роботи градієнтного бустинга. Як функцію витрат візьмемо середньоквадратичну помилку:

$$\text{Loss} = \text{MSE} = \sum_i \left(y_i - y_i^p\right)^2$$

де $y_i$ – і-те істинне значення цільової змінної, $y_i^p$ – прогнозне значення.

Наша мета мінімізувати функцію витрат. За для цього скористаємося градієнтними методами і розраховуючи нові передбачення, шукаємо значення, на яких MSE мінімальна.

$$y_i^p = y_i^p + \alpha * \frac{\delta \sum_i \left(y_i - y_i^p\right)^2}{\delta y_i^p}$$

$$y_i^p = y_i^p - \alpha * 2 * \sum_i \left(y_i - y_i^p\right)$$

де $\alpha$ – шаг навчання.

Отже, ми просто оновлюємо передбачення таким чином, що середньоквадратичну помилку прямувала до нуля і результати роботи моделі були максимально наближені до реальних.

### 2.5.1 Переваги і недоліки градієнтного бустингу

Переваги:

- Висока точність прогнозування. Дуже часто вища ніж у випадкового лісу;
- Стійкий до перенавчання;
- Не потребує попередньої очистки та налаштування даних. Деякі реалізації цього методі можуть працювати використовуючи не тільки числові дані.

Недоліки:

- Метод чутливий до викидів в даних. Адже кожне наступне дерево рішень буде намагатися зменшити похибку минулого;
- Неможливо масштабувати;
- Багато часу займає начання моделі;
- Велика кількість параметрів для налаштування.

## 2.6 Висновки до розділу 2

У цьому розділі були розглянуті основні методи машинного навчання, та методи протидії перенавчанню.

Лінійна модель вважається першим алгоритмом, який потрібно спробувати, він гарно працює для великих наборів данних, підходить для даних з великою розмірністю.

Дерево рішень дуже швидкий метод машинного навчання. Його результати можна візуалізувати.

Випадковий ліс майже завжди працює краще, чим одне дерево рішень. Це дуже стійкий та потужний метод. Погано працює з даними великої розмірності.

Градієнтний бустинг дерев рішень як правило більш точний ніж випадковий ліс. На відмінну від випадкового лісу повільніше навчається, проте швидкість передбачення вища та потребує менше пам’яті. Потребує порівняно з випадковим лісом налаштування більшої кількості параметрів.

Виходячи з того, що швидкодія та точність моделі є критично важливими параметрами далі в роботі розглядається різні налаштування градієнтного бустингу над деревами рішення, а саме реалізація пакету Catboost.

# РОЗДІЛ 3 РОЗРОБКА КОМП'ЮТЕРНОЇ ПРОГРАМИ ТА ПРАКТИЧНІ РЕЗУЛЬТАТИ ВИКОРИСТАННЯ МЕТОДІВ МАШИННОГО НАВЧАННЯ

## 3.1 Вибір програмного продукту

Програмний продукт було розроблено для використання в віртуальному середовищі Docker. В ході розробки програмного продукту були визначені можливі платформи для її реалізації: Java, C#, R, C++, Scala, Python. Кінцевий програмний продукт розроблено на мові програмування Python, використовуючи платформу Jupyter Notebook.

Python є загальноприйнятою мовою програмування при аналізі даних та розробці моделей машинного навчання, адже має велику кількість вбудованих пакетів програмних продуктів та має зручний синтаксис для освоєння. В Python є бібліотеки для візуалізації, кластеризації, загрузці даних, статистичних обчислень, обробки мови, зображень та багато чого іншого. Цей великий набір інструментів пропонує спеціалістам по роботі з даними великий набір інструментів широкого та вузького спрямування. Одним з найголовніших переваг цієї мови програмування є можливість прямої роботи з кодом через термінал або інструменти, як наприклад Jupyter Notebook.

Jupyter Notebook представляє собою інтерактивне вередовище для запуку програмного коду в браузері. Він дозволяє мати все в одному місці, графіки, дані, документацію та код, як також підтримує синтаксис спрощеної розмітки Markdown та потужний синтаксис LaTeX.

Програма складається з таких бібліотек, як pandas, numpy, matplotlib, catboost, datetime, sklearn.

## 3.2 Опис набору даних

Для розробки програмного продукту був використаний набір даних, який містить 2 000 000 записів про користувачів, які були відомі до спроби виконати рекурентний платіж і для кожного запису є данні про те, чи був він вдалий.

Маємо на вхід базу даних, що має 25 змінних, а саме:

- gender – стать користувача;
- age – вік користувача;
- app – пристрій користувача;
- country_group – до якої групи країн належить країна користувача;
- retry_number – спроба виконати данний рекурентний платіж;
- tariff – тарифний план користувача;
- gross – величина рекурентного платежу;
- type_id – тип тарифного плану;
- days_after_reg – скільки пройшло днів після реєстрації користувача;
- discount – знижка;
- descriptor – плтіжний шлюз;
- merchant_id – id мерчанту;
- payment_type – тип оплати;
- context_id – налаштування контексту;
- card_issuer – банк який віпустив картку користувача;
- pay_app – пристрій оплати користувача;
- card_brand – платіжна система;
- card_type – тип картки;
- sessions_in_5_days_before_rec – скільки було сесій у користувача перед здійсненням платежу;

- success_orders_before_rec – успішних сплат користувача на момент спроби списання коштів;
- cpa – ціна за яку був залучений користувач;
- type_trafic – тип трафіку з якого був залучен користувач;
- previous_response – результат минулої спроби списати кошти;
- is_success – результат.

Також датасет мав інші данні але вони були видалені в зв’язку з великою кількістю пропущених значень.

## 3.3 Чисельна оцінка якості моделі

Задача розрахунку вірогідності є частиною задачу класифікації, отже можна використовувати для оцінки якості моделі – confusion matrix (матриця помилок)

Матриця помилок – це прямокутна матриця котра показує кількість правдиво позитивних, правдиво негативних, хибно позитивних, хибно негативних прогнозів класифікатора, як зображено в табл 3

Таблиця 3.1 – Матриця помилок класифікації

| | y = 1 | y = 0 |
|---|---|---|
| $\hat{y} = 1$ | Правдиво позитивні (TP) | Хибно позитивні (FP) |
| $\hat{y} = 0$ | Хибно негативні (FN) | Правдиво негативні (TN) |

де $\hat{y}$ – це прогноз алгоритму;

y – істинний результат.

Отже, хибні результати моделі можуть приймати два варіанти: False Negative (FN) і False Positive (FP).

Найпростішими та найпоширенішими оцінками якості класифікації є – помилка (ERR) та вірність (ACC) прогнозу.

$$ERR = \frac{FP + FN}{FP+FN+TP+TN}$$

$$ACC = \frac{TP + TN}{FP + FN + TP + TN}$$

Дані показники представляють загальну інформацію про те, яка кількість даних правильно та не правильно класифіковано. Проте ці показники не показують всю повноту моделі при незбалансованому датасеті. В таких випадках використовую для аналізу – точність (PRE) та повноту (REC). Вони показують наскульки точним є позитивний прогноз та частку «знайдених» істинних міток.

$$PRE = \frac{TP}{FP + TP}$$

$$REC = \frac{TP}{FN + TP}$$

На практиці часто використовують комбінацію точності та повноти в вигляді метрики F1:

$$F1 = \frac{PRE \times REC}{PRE + REC}$$

Багато уваги в роботі приділено оцінці якості моделі завдяки ROC-кривій, вона дозволяє візуально зобразити точність моделі. Площа під ROC-кривою має назву ROC-AUC (area under curve), вона використовується як характеристика моделі.

ROC-крива – це широко вживаний інструмент для відбору моделей для класифікації, базуючись на їх якості відносно долі хибно позитивних та істинно позитивних результатів, котрі розраховуються шляхом зміщення порогу класифікації.

Звертаючи увагу на те, що значення вірогідностей будуть використовуватися для прогнозування прибутку та подальшого використання цих значень при оптимізації рекламних компаній, модель будемо також оцінювати по тому наскільки вірогідний прибуток наближений до істинного для кожного з каналів трафіку.

## 3.4 Catboost

CatBoost – це метод машинного навчання, заснований на градієнтному бустінгу над деревами прийняття рішень, розроблений компанією.

Основні переваги CatBoost:

- Чудова якість порівняно з іншими бібліотеками реалізацій градієнтного бустингу над деревами рішення у багатьох наборах даних;
- Найвища швидкість прогнозування в задачах класифікації;
- Підтримка як числових, так і категоричних ознак;
- Підтримка графічних адапторів (GPU);
- Наявні інструменти візуалізації.[6]

## 3.5 Результати роботи моделей

Для початку була взята модель CatBoostClassifier з початковими налаштуваннями. Модель навчалася на 70% даних та оцінювалася на інших 30%

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.87 | 0.98 | 0.92 | 241309 |
| 1 | 0.59 | 0.16 | 0.26 | 42314 |
| accuracy | | | 0.86 | 283623 |
| macro avg | 0.73 | 0.57 | 0.59 | 283623 |
| weighted avg | 0.83 | 0.86 | 0.82 | 283623 |

confusion_matrix:

--6895--4850
--35419--236459

Рисунок 3.1 – Оцінка початкової моделі

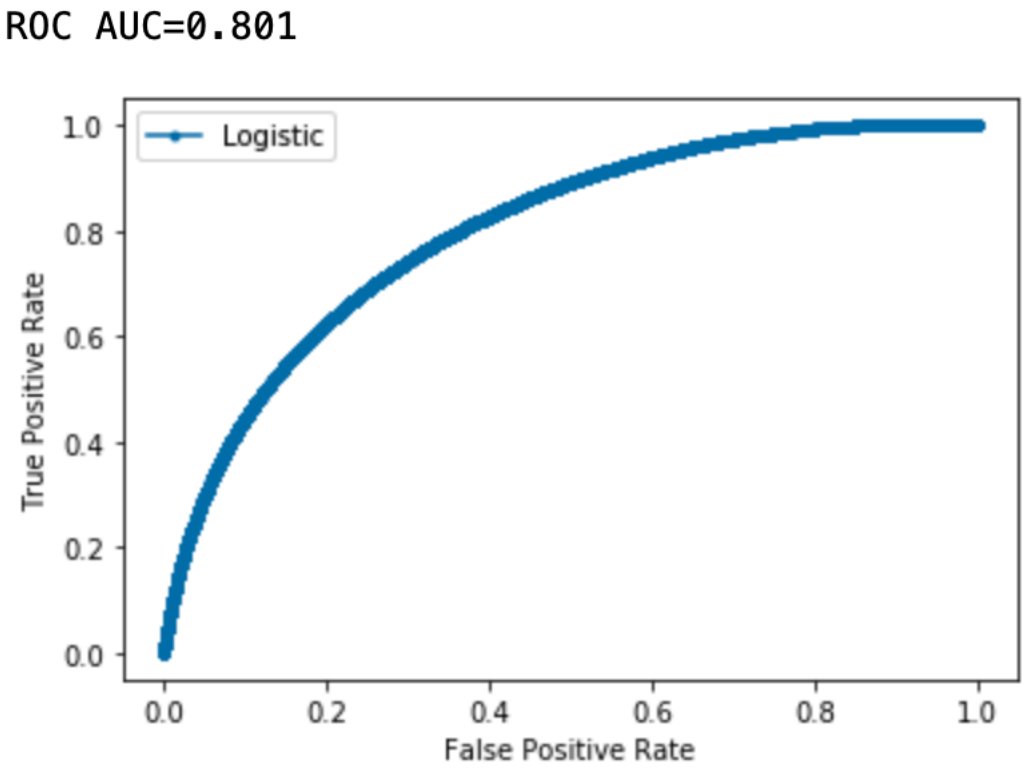



Рисунок 3.2 – ROC-крива моделі та значення ROC-AUC

На рис 3.1 та рис 3.2 зображені основні метрики оцінки якості моделі

Для оцінки точності прогнозування прибутку розрахуємо відхилення прогнозу від істинного значення.

$$DEV = 7.3249\%$$

Результат дає нам зрозуміти, що наша модель дещо завищує в загалом вірогідності. Для більшого розуміння поведінки моделі виведемо розподіл похибки на рис 3.4

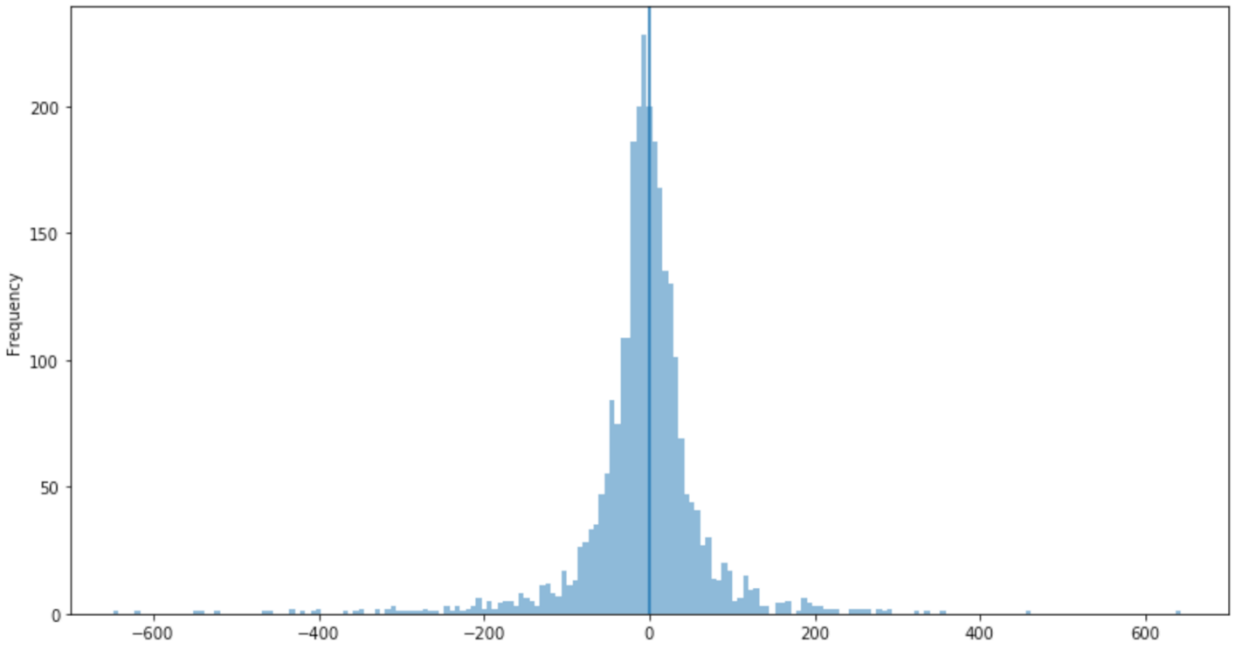



Рисунок 3.4 – Розподіл похибки

Розподіл похибки досить схожий на нормальний розподіл. Проте видно що дещо більша кількість похибок лежить зліва від 0, що означає, що модель завищує значення ймовірностей.

Для оцінки точності моделі використаємо метрики MAE (mean absolute error) та RMSE (root-mean-square error):

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}}$$

$$\mathrm{MAE} = \frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|}{n}$$

Значення цих метрик можна побачити на рис 3.5

```
MAE: 54.34562234
RMSE: 109.208438
```

Рисунок 3.5 – MSE, RMSE

Для оцінки якості моделі у відносних значеннях будемо використовувати метрику MAPE (mean absolute percentage error):

$$\text{MAPE} = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{y_t - \hat{y}_t}{y_t}\right| * 100$$

Проте зважаючи на те, що значення істинних прибутків істотно відрізняється один від одного, та зважаючи на предмтну область, будемо використовувати weighted MAPE

$$\text{weighted MAPE} = \frac{\sum_{t=1}^{n}|y_t - \hat{y}_t|}{\sum_{t=1}^{n} y_t} * 100$$

Результати можна побачити на рисунку 3.6

```
MAPE: 45.34%
w-MAPE: 21.46%
```

Рисунок 3.6 – MAPE, weighted MAPE

Для розуміння причини такої великої різниці виведемо розподіл істинних значень прибутків на рисуснку 3.7

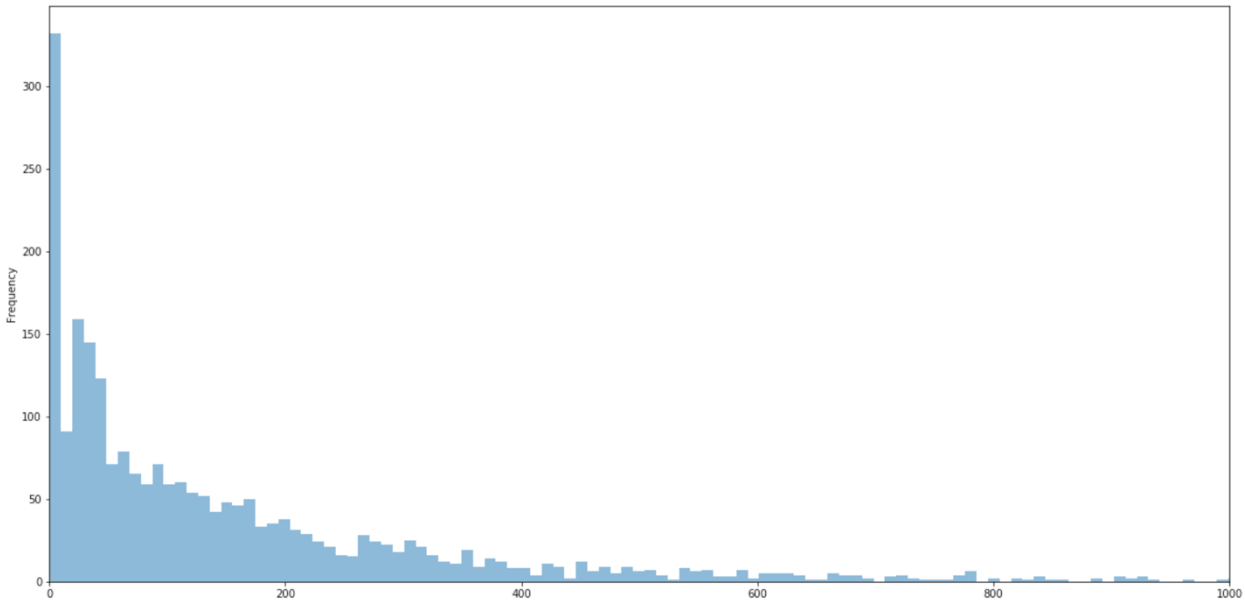



Рисунок 3.7 – Розподіл прибутків

Отже, можна зробити висновок, що відхилення при низькому істинному значені прибутку сильно впливають на оцінку. Далі в роботі будемо використовувати лише weighted MAPE.

Скористуємося вбудованою функцією get_feature_importance для того, щоб визначити які саме змінні найбільше всього впливають на передбачення (рисунок 3.8).

```
model.get_feature_importance(prettified=True)
```

| | Feature Id | Importances |
|---|---|---|
| 0 | previous_response | 42.776385 |
| 1 | description | 21.669330 |
| 2 | success_orders_before_rec | 5.958408 |
| 3 | card_issuer | 3.984794 |
| 4 | net | 3.611474 |
| 5 | country_group | 3.145448 |
| 6 | retry_number | 3.104263 |
| 7 | gross | 2.738542 |
| 8 | days_after_reg | 2.594875 |
| 9 | sessions_in_5_days_before_rec | 2.134494 |
| 10 | card_type | 1.864427 |
| 11 | payment_type | 1.338369 |
| 12 | age | 1.038223 |
| 13 | card_brand | 1.017631 |
| 14 | cpa_subchannel | 0.486234 |
| 15 | gender | 0.485108 |
| 16 | context_id | 0.440741 |
| 17 | descriptor | 0.428107 |
| 18 | discount | 0.274982 |
| 19 | pay_app | 0.243623 |
| 20 | type_trafic | 0.178240 |
| 21 | merchant_id | 0.174210 |
| 22 | tariff | 0.143628 |
| 23 | country_code | 0.134638 |
| 24 | app | 0.033827 |

Рисунок 3.8 – Вага параметрів при розрахунку передбачення

Для прискорення навчання моделі видалимо ті змінні, вага яких менше 1 відсотка та скористаємося методом випадкового пошуку гіперпараметрів. В таблиці 3.2 наведеній результати.

Таблиця 3.2 – Таблиця результатів

| | Параметри моделі | | | | Метрики | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | reg | subsample | rsm | depth | auc | Dev | MAE | RMSE | w-MAPE |
| 1 | 4 | 0.7 | 0.85 | 4 | 0.79381 | 6.77% | 51.587 | 95.123 | 17.219 |
| 2 | 4 | 0.7 | 0.85 | 12 | 0.80089 | 5.69% | 48.788 | 88.555 | 16.259 |
| 3 | 4 | 0.7 | 1 | 4 | 0.79356 | 6.72% | 51.681 | 95.587 | 17.258 |
| 4 | 4 | 0.7 | 1 | 12 | 0.80078 | 5.60% | 48.579 | 87.512 | 16.170 |
| 5 | 4 | 0.85 | 0.85 | 4 | 0.79355 | 5.80% | 50.634 | 92.396 | 16.891 |
| 6 | 4 | 0.85 | 0.85 | 12 | 0.80070 | 5.93% | 49.118 | 88.950 | 16.357 |
| 7 | 4 | 0.85 | 1 | 4 | 0.79391 | 6.37% | 51.173 | 94.089 | 17.083 |
| 8 | 4 | 0.85 | 1 | 12 | 0.80106 | 5.33% | 48.221 | 86.632 | 16.050 |
| 9 | 7 | 0.7 | 0.85 | 4 | 0.79320 | 7.09% | 52.091 | 96.202 | 17.391 |
| 10 | 7 | 0.7 | 0.85 | 12 | 0.80092 | 5.52% | 48.472 | 87.253 | 16.132 |
| 11 | 7 | 0.7 | 1 | 4 | 0.79383 | 6.35% | 51.064 | 93.903 | 17.047 |
| 12 | 7 | 0.7 | 1 | 12 | 0.80019 | 5.78% | 48.914 | 87.847 | 16.265 |
| 13 | 7 | 0.85 | 0.85 | 4 | 0.79379 | 6.15% | 50.837 | 93.037 | 16.959 |
| 14 | 7 | 0.85 | 0.85 | 12 | 0.80040 | 5.99% | 49.078 | 89.260 | 16.343 |
| 15 | 7 | 0.85 | 1 | 4 | 0.79338 | 6.60% | 51.539 | 95.116 | 17.210 |
| 16 | 7 | 0.85 | 1 | 12 | 0.80069 | 5.39% | 48.378 | 86.627 | 16.091 |

Можна зробити висновок, що точність моделі залежить від глибини дерева, а саме, чим глибше, тим точніше модель

## 3.6 Висновки до розділу 3

В ході виконання роботи над розробкою статистичної моделі для підрахунку ймовірностей відхилення рекурентних платежів було виконано дослідження з надходження найкращого методу машинного навчання та підбір його найкращих гіперпараметрів за допомогою методі випадкового пошуку. Розглянуто декілька метрик та їй формули для оцінки точності прогнозування моделі. Найбільшу точність показала модель градієнтного бустингу з великою глибиною дерев рішень.

# РОЗДІЛ 4 ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ

## 4.1 Постановка задачі техніко-економічного аналізу

У роботі застосовується метод функціонально-вартісного аналізу для проведення техніко-економічного аналізу розробки програмного продукту для розрахунку вірогідності відхилення рекурентних платежів.

Технічні вимоги до продукту наступні:

- програмний продукт повинен функціонувати на будь-якій операційній системі, незалежно від апаратного забезпечення та її конфігурації;
- програмний продукт повинен мати високу швидкість розрахунку;
- програмний продукт повинен бути відказостійким;
- програмний продукт повинен забезпечувати максимальну відсутність похибок;
- програмний продукт повинен бути зручним та зрозумілим для використання;
- програмний продукт повинен мати можливість інтеграції з усіма можливими системами прийняття рішень та поєднання з іншими програмними системами;
- передбачати мінімальні витрати на впровадження програмного продукту.

## 4.2 Обґрунтування функцій програмного продукту

Функція F0 — розробка програмного продукту, що приймає на вхід дані та виводить вірогідності відхилення рекурентних платежів. Виходячи з цього, можна виділити наступні основні функції ПП:

$F_1$— вибір мови програмування:

а) мова програмування Python;

б) мова програмування C++.

$F_2$— вибір алгоритму вирішення:

а) загальні методи розрахунку;

б) методи машинного навчання.

$F_3$— збереження результатів роботи:

а) агреговано по каналам трафіку;

б) окремо для кожного рекурентного платежу.

Побудуємо морфологічну карту за даними варіантами (рисунок 4.1).

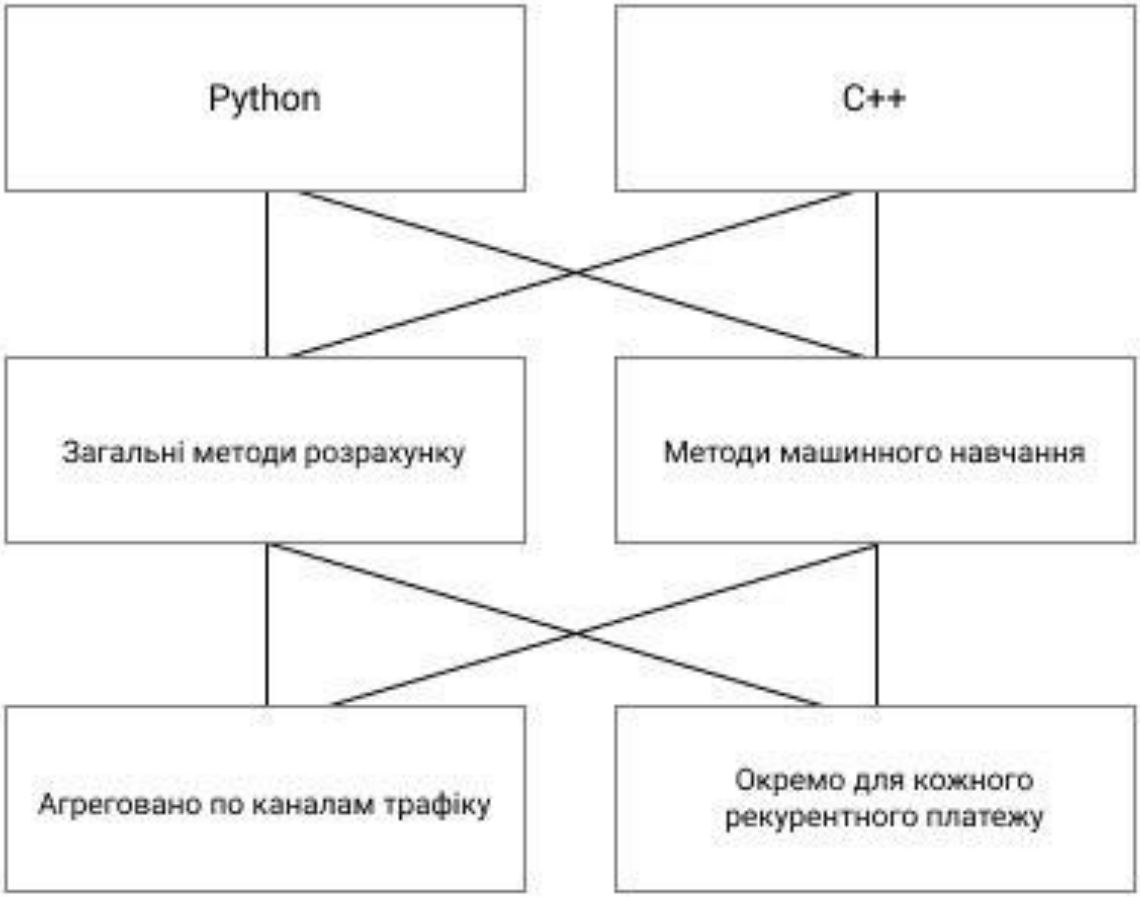



Рисунок 4.1 - Морфологічна карта

Позитивно-негативна матриця варіантів основних функцій (таблиця 4.1)

Таблиця 4.1 – Позитивно-негативна матриця

| Основні функції | Варіанти реалізації | Переваги | Недоліки |
|---|---|---|---|
| F1 | А | Висока швидкість роботи продукту | Складність розробки |
| | Б | Нижча вартість, відносна легкість проведення дослідів | Низька швидкодія |
| F2 | А | Швидка та зрозуміла реалізація | Низька точність |
| | Б | Висока точність роботи | Час на розробку |
| F3 | А | Малий обсягу пам’яті для зберігання | Недоступність результатів попередніх тестувань |
| | Б | Можливість більш глибокого аналізу | Великий обсяг пам’яті для зберігання |

За аналізом позитивно-негативної матриці можна зробити висновок, що при розробці програмного продукту деякі варіанти реалізації функцій потрібно відкинути, бо вони не відповідають поставленій перед програмним продуктом меті.

Функція F1:

Оскільки Python дозволяє просто та швидко працювати з даними, то розглядаємо лише цей варіант.

Функція F2:

Точність розрахунку критично важлива тому варіант а) не розглядається

Функція F3:

Так як дані у сховищах можуть міститися у різних виглядах, тому розглядаються обидві варіанти.

Таким чином, будемо розглядати такий варіант реалізації ПП:

а) F1а — F2б — F3а;

б) F1а — F2б — F3б.

## 4.3 Обгрунтування системи параметрів

Для оцінювання якості розглянутих реалізацій обрані параметри, описані нижче.

Для визначення оцінки реалізацій виберемо параметри (опис наведено в таблиці 4.2).

Таблиця 4.2 – Основні параметри програми

| Назва параметра | Умовні позначення | Одиниці виміру | Значення параметра | | |
|---|---|---|---|---|---|
| | | | гірші | середні | кращі |
| Час на реалізацію програми | X1 | год | 80 | 65 | 45 |
| Час на обробку результату | X2 | с | 10 | 6 | 3 |
| Потенційний об’єм програмного коду | X3 | год | 2000 | 1300 | 900 |

За значеннями в таблиці значеннями побудуємо графічні характеристики параметрів (рисунок 4.2, рисунок 4.3, рисунок 4.4)

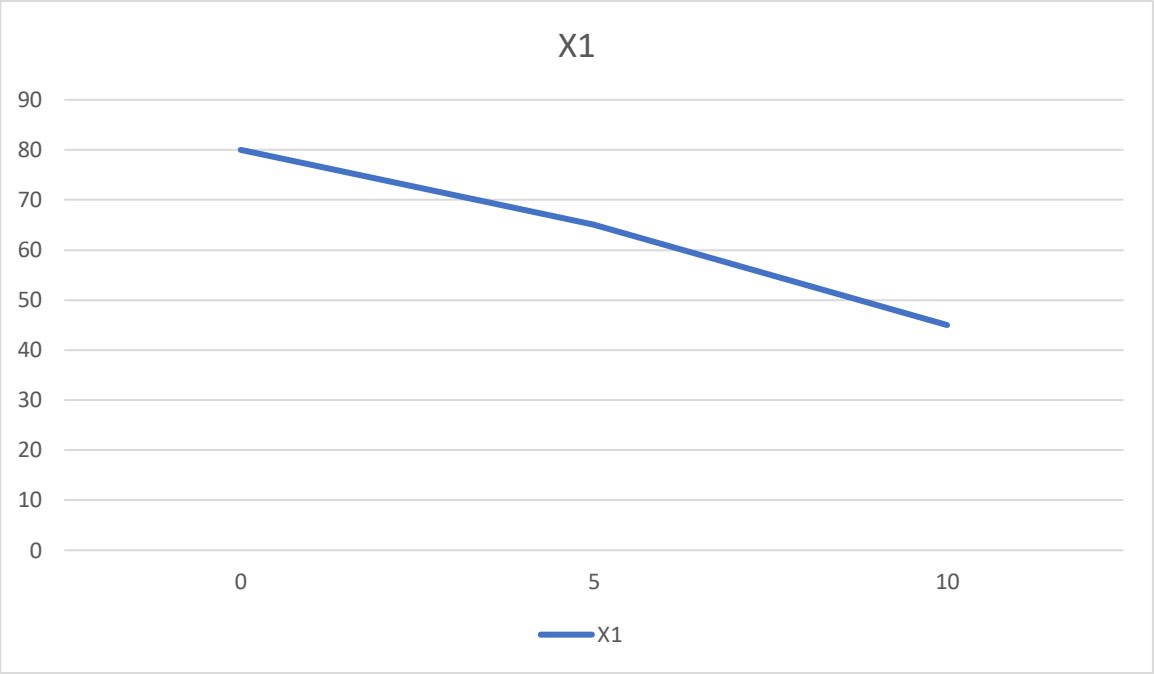



Рисунок 4.2 – параметр X1

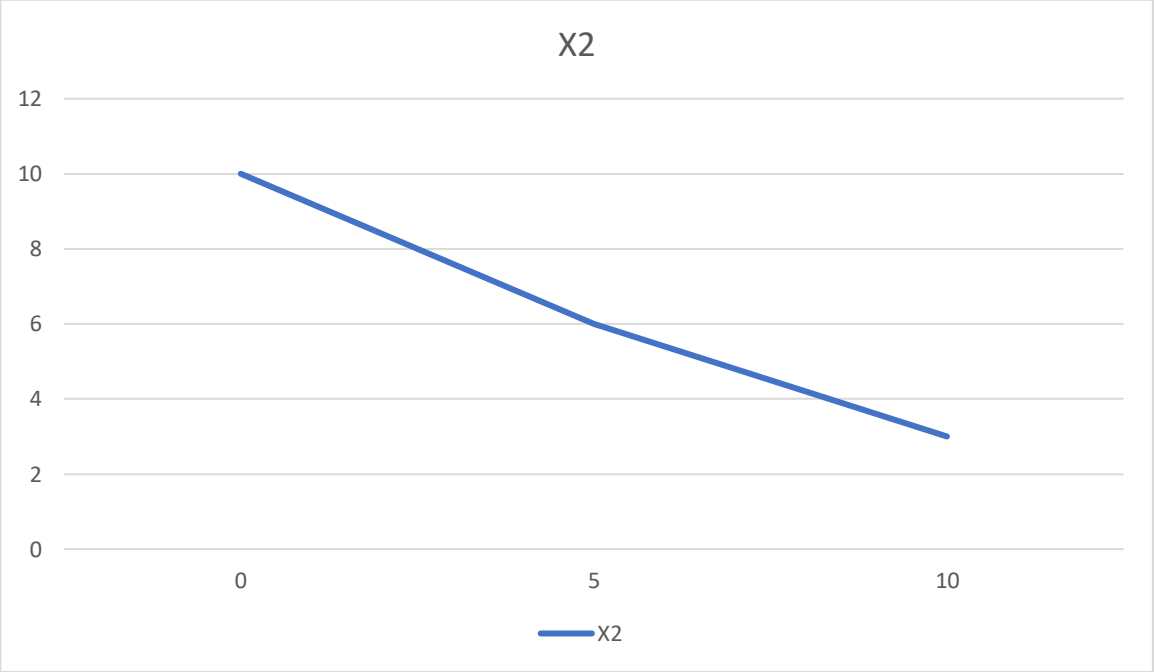



Рисунок 4.3 – параметр X2

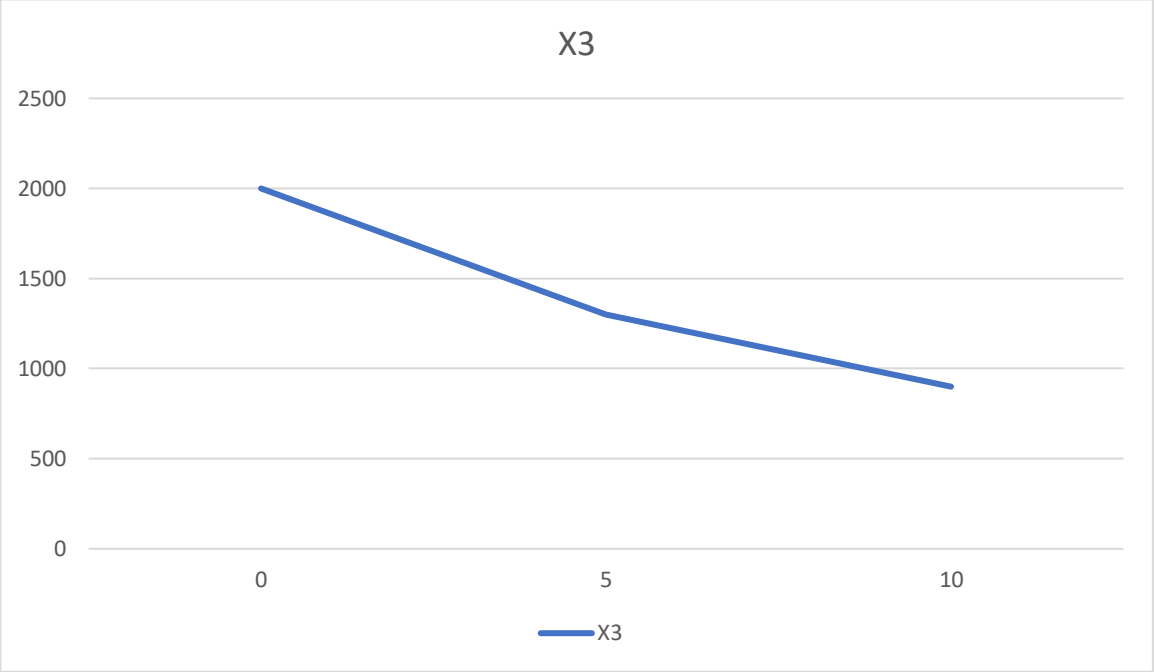



Рисунок 4.4 – параметр X3

Результати експертного ранжування наведені у таблиці 3.

Таблиця 4.3 – Результати ранжування параметрів

| Параметр | Назва параметра | Одиниці виміру | Ранг параметра за оцінкою експерта | | | | | | | Сума рангів $R_i$ | Відхилення $\Delta_i$ | $\Delta_i^2$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | | | |
| X1 | Час на реалізацію програми | год | 2 | 2 | 1 | 2 | 2 | 1 | 2 | 12 | -2 | 4 |
| X2 | Час на обробку результату | год | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 21 | 7 | 49 |
| X3 | Потенційний об’єм програмного коду | год | 1 | 1 | 2 | 1 | 1 | 2 | 1 | 9 | -5 | 25 |
| | Разом | | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 42 | 0 | 78 |

Найбільшим вважаємо ранг 3, відповідно найменшим є ранг 1. Чим більший ранг експерт присвоює параметру, тим важливішим він, або вона, його вважає.

Коефіцієнт узгодженості дорівнює:

$$W = \frac{12S}{N^2(n^3-n)} = \frac{12*78}{49(27-3)} \approx 0.796>0.67$$

Проведено попарне порівняння всіх параметрів. Результати в таблиці 4.4

Таблиця 4.4 – Попарне порівняння параметрів

| Параметри | Експерти | | | | | | | Кінцева оцінка | Числове значення |
|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | | |
| X1 і X2 | < | < | < | < | < | < | < | < | 0.5 |
| X2 і X3 | > | > | > | > | > | > | > | > | 1.5 |
| X1 і X3 | > | > | < | > | > | < | > | > | 1.5 |

За результатами попарного порівняння обчислюємо вагомість кожного з критеріїв (таблиця 4.5 )

Таблиця 4.5 – Розрахунок вагомості параметрів

| Параметри $x_i$ | Параметри $x_j$ | | | Перша ітерація | | Друга ітерація | |
|---|---|---|---|---|---|---|---|
| | X1 | X2 | X3 | $b_i$ | $K_{bi}$ | $b_i^1$ | $K_{bi}$ |
| X1 | 1 | 0.5 | 1.5 | 3 | 0.333 | 8 | 0.32 |
| X2 | 1.5 | 1 | 1.5 | 4 | 0.444 | 11.5 | 0.46 |
| X3 | 0.5 | 0.5 | 1 | 2 | 0.222 | 5.5 | 0.22 |
| Всього: | | | | 9 | 1 | 25 | 1 |

## 4.4 Аналіз варіантів реалізації функцій

Таблиця 4.6 – Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

| Основні функції | Варіант реалізації функції | Параметри | Абсолютне значення параметра | Бальна оцінка параметра | Коефіцієнтвагомостіпараметра | Коефіцієнтрівняякості |
|---|---|---|---|---|---|---|
| F1 | А | X1 | 70 | 7 | 0.32 | 2.24 |
| | | X3 | 1100 | 4 | 0.22 | 0.88 |
| F2 | Б | X1 | 50 | 4 | 0.32 | 1.28 |
| F2 | А | X1 | 65 | 5 | 0.32 | 1.6 |
| | | X2 | 6 | 5 | 0.46 | 2.3 |
| | Б | X1 | 40 | 3 | 0.32 | 0.96 |
| | | X2 | 7 | 6 | 0.46 | 2.76 |

Отже рівень якості для варіантів складе

К1 = 2.24+ 0.88+1.28+ 1.6+ 2.3= 8.3

К2 = 2.24+ 0.88+1.28+ 0.96+ 2.76= 8.12

З результатів випливає, що перший варіант є кращим.

## 4.5 Економічний аналіз варіантів розробки програми

Для оцінки вартості розробки проведемо розрахунок трудомісткості. Обидва варіанти мають такі основні завдання:

1) Розробка моделі
2) Розробка програмного продукту

Окрім цього кожен з варіантів має додаткове завдання (3.1. – для першого, 3.2 – для другого)

3.1) Освоєння теоретичної бази для роботи з моделью

3.2) Фільтрування даних для створення моделі

За складністю алгоритмів до групи 1 відносять завдання 1 та 2, до групи складності 2 – 3.2, до групи складності 3 – 3.1. За степенем новизни до групи А належить завдання 2, до групи Б – 3.2, до групи В - 1 та 3.1. Спираючись на норми розрахункового часу визначимо трудомісткість.

Для завдання 1: (алгоритм групи складності 1, ступінь новизни В, вид використаної інформації – НДІ)ТР = 43 людино-днів, $К_п$= 0.81, $К_{ск} = 1$, $К_{ст} = 0.7$. Тому Т1 = 43*0.81*0.7= 24.38 людино дні.

Для завдання 2: (алгоритм групи складності 1, ступінь новизни А, вид використаної інформації – НДІ)ТР = 90 людино-днів, $К_п$= 1.7, $К_{ск} = 1$, $К_{ст} = 0.7$. Тому Т2 = 90*1.7*0.7= 107.1 людино дні.

Для додаткового завдання 3.1: (алгоритм групи складності 3, ступінь новизни В, вид використаної інформації НДІ) ТР =12 людино-днів, КП =0.6, КСК = 1, КСТ = 0.7, КСТ.М =1. Тому Т31 = 12*06*0.7 = 5.04 людино-днів.

Для додаткового завдання 3.2: (алгоритм групи складності 2, ступінь новизни Б, вид використаної інформації НДІ) ТР =27 людино-днів, КП =1.26, КСК = 1, КСТ = 0.7, КСТ.М = 1. Тому Т32 = 27*1.26*0.7 = 23.81 людино-днів.

Звідси виходить, що повна трудомісткість першого варіанту

$$T = 24.38 + 107.1 + 5.04 = 136.52$$

А повна трудомісткість другого варіанту

$$T = 24.38 + 107.1 + 23.81 = 155.29$$

В розробці мають брати участь один програміст з окладом 16 000 грн та один аналітик з окладом 12 000. Погодинна заробітна плата працівників складе:

$$C = (16000 + 12000)/2*21*8 = 83.33 \text{ грн}$$

Поваріантно:

$$1)\ C1 = 136.52*83.33*1.2 = 13651.454 \text{ грн}$$

$$2)\ C2 = 155.29*83.33*1.2 = 15528.379 \text{ грн}$$

Відрахування ЄСВ складають 22%, або, поваріантно:

$$C' = C1*0.22 = 13651.454*0.22 = 3003.319 \text{ грн}$$

$$C' = C2*0.22 = 15528.379 *0.22 = 3416.243 \text{ грн.}$$

Визначаємо витрати на оплату однієї машино-години. З урахуванням заробітної плати програміста в розмірі 16000 грн з коефіцієнтом зайнятості 0.2, отримуємо:

$$Сг = 12*М*Кз = 12*16000*0.2 = 38400 \text{ грн}$$

Включаючи додаткову плату:

$$Сзп = Сг * (1 + Кз) = 38400 * 1.2 = 46080 \text{ грн}$$

Відповідно, відрахування на соціальний внесок складуть

$$Свід = Сзп * 0.22 = 46080 *0.22 = 10137.6 \text{ грн}$$

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 25000 грн.

$$Са = Ктм * Ка * Цпр = 1.15 * 0.25 * 25000 = 7187.5 \text{ грн}$$

Профілактичі трати $Ср = Ктм*Кр*Цпр = 1.15*0.05*25000 = 1437.5$ грн

Ефективний годинний фонд:

$$ТЕФ = (ДК – ДВ – ДС – ДР) \cdot tЗ \cdot КВ = (365 – 104 – 8 – 16) \cdot 8 \cdot 0,9 = 1706.4 \text{ годин.}$$

Звідси виходить, що витрати на оплату електроенергії, з урахуванням ПДВ, складуть:

$$СЕЛ = 1706.4 * 0,6*0,5 * 1.75= 873.49 \text{ грн}$$

Накладні витрати складуть:

$$Сн = Цпр*0.67 = 25000 *0.67 = 16755 грн$$

Річні витрати на експлуатацію:

$$СЕКС =СЗП+ СВІД+ СА + СР+ СЕЛ + СН=46080 + 10137.6 +7187.5 + 1437.5 +873.49 +16755 = 82471.09 грн$$

Тоді собівартість однієї машино-години складе:

$$СМ-Г = СЕКС/ ТЕФ=82471.09 /1706.4= 48.33 грн/год$$

Тоді витрати на оплату машинного часу складають:

$$См = 48.33 * 136.52= 6598.0116$$
$$См =48.33 * 155.29= 7505.1657$$

Накладні витрати становитимуть 67% від заробітної плати:

$$0.67*13651.454 =9146.474 для першого варіанту$$
$$0.67*15528.379 = 10404.0139 для другого для другого варіанту$$

Таким чином, вартість розробки ПП та проведення дослідів складає:

$$СПП = СЗП+ СВІД+ СМ +СН$$

Для першого:

$$13651.454 + 3003.319 +6598.0116+ 9146.474 = 32399.2586$$

Для другого:

15528.379 + 3416.243 + 7505.1657+10404.0139 = 36853.8016

## 4.6 Вибір кращого варіанту ПП техніко-економічного рівня

Коефіцієнти техніко-економічного рівня, розраховані за формулою:

$$КТЕРj = ККj / СФj$$

Для першого варіанту реалізації:

Ктер1 = 8.3/29219.0806=28.41*10^(-5)

Для другого варіанту реалізації:

Ктер2 = 8.12 /30998.3442=26.19*10^(-5)

## 4.7 Висновки до розділу 4

Отже, враховуючи всі дослідження, які ми провели, можна сказати, що другий варінт реалізації є найбільш оптимальним з розглянутої точки зору Цьому варіанту продукту відповідають такі параметри:

- мова програмування: Python;

- методи машинного навчання;
- окремо для кожного рекурентного платежу.

При реалізації цієї версії продукту, всі поставлені вимоги будуть виконані оптимальним способом.

# ВИСНОВКИ

Дана робота присвячена  аналізу та побудові моделей розрахунку ймовірностей відхилення рекурентних платежів для подальшого використання результатів роботи моделі у прийнятті рішень при оптимізації маркетингових кампаній. Було розроблено і проаналізовано основні моделі машинного навчання.

У першому розділі було визначено актуальність даної теми та надано основні визначення з цієї проблематики, а саме, що таке бізнес-модель та які є види, що таке рекурентний платіж, які є причини його відхилення, та як відбувається процес обробки рекурентного платежу.

У другому розділі надано огляд основних методів машинного навчання для підрахунку ймовірностей, способи уникнення перенавчання, а такж які є нюанси у кожної з моделі. Був обраний градієнтний бустинг над деревами рішення, завдяки його високій точності та швидкості підрахунку.

Третій розділ присвячено результатам аналізу даної проблеми на мові програмування Python у програмному середовищі Jupyter Notebook р використанням пакету Catboost. Показана робота програми та аналіз метрик для оцінювання точності моделей. Наведено таблицю, де наведені основні метрики для кожного набору гіперпараметрів, які були вибрані за допомогою випадкового пошуку. Був проведений аналіз цієї таблиці та обраний найкращий варіант гіперпараметрів: reg=7, subsample=0.85, rsm=1, depth=12, який показав найбільшу точність: ROC-AUC = 0.80069, dev=5.39%, MAE = 48.378, RMSE=86.627, w-MAPE=16.091.

Наступними кроками покращення створеного програмного продукту є підрахунок не тільки запланованих рекурентів, а й тих, які можливо будуть у майбутньому.

# ПЕРЕЛІК ПОСИЛАНЬ


1. Carol M. Kopp. What is business model? URL: https://www.investopedia.com/terms/b/businessmodel.asp (дата звернення 01.05.2020).
2. Freemium. URL: https://uk.moneycash4u.com/freemium (дата звернення 01.05.2020).
3. Heidenreich H. What are the types of machine learning?. 2018. URL: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53 (дата звернення: 10.04.2020).
4. Nahla Ben Amora, Salem Benferhat, Zied Elouedia. Qualitative Classification with Possibilistic Decision Trees, 2006. URL: https://www.sciencedirect.com/science/article/pii/B9780444520753500145 (дата звернення: 20.05.2020).
5. Great Learning Team. 2020. URL: https://www.mygreatlearning.com/blog/decision-tree-algorithm/
6. Catboost. URL: https://catboost.ai (дата звернення: 20.05.2020).

# ДОДАТОК А ТЕКСТ ПРОГРАМИ

```
import pandas as pd
import vertica_python
import numpy as np
import time
import warnings
import json

from matplotlib import pyplot as plt
from card_identifier.card_issuer import identify_card_issuer
from sklearn.datasets import make_classification
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import roc_curve, classification_report, roc_auc_score,
confusion_matrix, mean_absolute_error as mae, mean_squared_error as mse
from matplotlib import pyplot
from catboost import CatBoostClassifier, Pool
from sklearn.model_selection import StratifiedShuffleSplit
from datetime import datetime

warnings.filterwarnings("ignore")

!jupyter nbextension enable --py widgetsnbextension

%load_ext autoreload
```

```
%autoreload 2



def sberbank_detector(x):
    if 'bpssberbank' in x.lower():
        return 'by-sberbank'
    elif 'sberbank' in x.lower():
        return 'rus-sberbank'
    else:
        return x



def none_detector(x):
    if x is None or x == '':
        return np.nan
    else:
        return x



with vertica_python.connect(**conn_vc) as connection:
     sql = pd.read_sql(script, connection, index_col='id')

sql.to_csv('dataset.csv')

sql_c = sql.copy()
```

```
sql = sql_c.copy()

with vertica_python.connect(**conn_vc) as connection:
    card_bin_matcher = pd.read_sql(card_bin_matcher_script, connection,
                                   index_col='card_bin').applymap(lambda x: none_detector(x))

card_bin_matcher.loc['220220', 'card_issuer_new'] = 'SBERBANK'

sql = pd.merge(sql,
               card_bin_matcher.reset_index(),
               on='card_bin',
               how='left')
print(' - ')
for column in ['card_issuer', 'card_type', 'card_brand']:
    rep = sql.loc[(sql[column].isna()) & (sql[f'{column}_new'].notna())]
    sql.loc[(sql[column].isna()) & (sql[f'{column}_new'].notna()), f'{column}'] =
rep.loc[:, f'{column}_new']
print(' - ')
sql.fillna({'card_brand': 'VISA',
            'card_issuer': 'some_bank',
            'card_type': 'DEBIT'}, inplace=True)
sql.drop(columns=[
    'card_bin', 'card_brand_new', 'card_issuer_new', 'card_type_new', 'cnt', 'net'],
inplace=True)
print(' - ')
```

```
sql.context_id = sql.context_id.apply(lambda x: str(x))
sql.tariff = sql.tariff.apply(lambda x: str(x))
sql.card_issuer = sql.card_issuer.apply(sberbank_detector)
df.to_csv('dataset.csv', index=False)

df = pd.read_csv('dataset.csv')
df.reg_dt = pd.to_datetime(df.reg_dt).apply(lambda x: x.date())
df.recur_dt = pd.to_datetime(df.recur_dt).apply(lambda x: x.date())
df.tariff = df.tariff.apply(lambda x: str(x))
df.type_id = df.type_id.apply(lambda x: str(x))
df.context_id = df.context_id.apply(lambda x: str(x))



df.type_id = df.type_id.apply(lambda x: str(x))

drop_columns = ['is_success','reg_dt','cpa_subchannel', 'gender', 'context_id',
'descriptor', 'discount',
            'pay_app', 'type_trafic', 'merchant_id', 'tariff', 'country_code',
             'app', 'channel_id', 'type_id',  'recur_dt', 'parent_order_id']



delimeter_date = '2019-11-30'

train = df[(df.days_after_reg < 61) & (df.reg_dt < pd.to_datetime(delimeter_date))
& ((df.tariff == '31') | ((df.tariff == '30') & (df.type_id == '3')))]
```

```
test = df[(df.days_after_reg < 61) & (df.reg_dt >= pd.to_datetime(delimeter_date))
& ((df.tariff == '31') | ((df.tariff == '30') & (df.type_id == '3')))]

X_train = train.drop(columns=drop_columns)
y_train = train.is_success

X_test = test.drop(columns=drop_columns)
y_test = test.is_success

print(f'train: \n{y_train.value_counts()}\n')
k =  y_train.value_counts().iloc[1] / y_train.value_counts().iloc[0]
print('%.3f' % (y_train.value_counts().iloc[1] / y_train.value_counts().iloc[0]))

print(f'\ntest: \n{y_test.value_counts()}')
print('\n%.3f' % (y_test.value_counts().iloc[1] / y_test.value_counts().iloc[0]), '\n')

train_drop = y_train[y_train == 0].sample(y_train.value_counts().loc[0] -
y_train.value_counts().loc[1]).index

X_train = X_train.drop(train_drop)
y_train = y_train.drop(train_drop)

print('train: {}\ntest: {}'.format(y_train.value_counts(), y_test.value_counts()))
```

```
mother_grid = {}
i = 0

for l2_leaf_reg in [4, 7]:
    for subsample in [0.7, 0.85]:
        for rsm in [0.85, 1]:
            for depth in [4, 12,]:
                mother_grid[i] = {
                    'l2_leaf_reg': l2_leaf_reg,
                    'subsample': subsample,
                    'rsm': rsm,
                    'depth': depth}
                i += 1

cat_features = X_train.columns[X_train.dtypes == np.object]

train_pool = Pool(
    data=X_train,
    label=y_train,
    weight=(y_train * (k - 1) + 1).values,
    cat_features=cat_features

)

test_pool = Pool(
```

```
    data=X_test,
    label=y_test,
    cat_features=cat_features
)



for i in range(len(mother_grid)):
    model = CatBoostClassifier(
                            learning_rate=0.1,
                           iterations=70,
                           custom_loss=['AUC', 'Accuracy', 'F1', 'Recall', 'Precision'],#,
train_dir='/tmp/',
                           use_best_model=True,
                            **mother_grid[i]
                          )
    model.fit(train_pool,eval_set=test_pool, verbose=0)
    preds_proba = model.predict_proba(X_test)

    lr_auc = roc_auc_score(y_test.values, preds_proba[:,1])

    tmp = test[['gross', 'reg_dt', 'recur_dt', 'is_success',  'channel_id']]
    tmp['actual_money'] = tmp.gross * tmp.is_success
    tmp['model_probability'] = preds_proba[:,1]
    tmp['predicted_money'] = tmp.gross * tmp.model_probability

    all_predicted_money = tmp.predicted_money.sum()
```

```
    res = tmp.groupby(['reg_dt',
'channel_id'])['actual_money','predicted_money'].sum().reset_index()
    mae_model = mae(res.actual_money, res.predicted_money)
    rmse_model = np.sqrt(mse(res.actual_money, res.predicted_money))

    r = res.drop(res[res.actual_money == 0].index)
    weighted_mape = sum(abs(r.actual_money - r.predicted_money)) /
sum(r.actual_money) * 100

    print(list(mother_grid[i].values()))
    print('%.5f'%lr_auc,f'{((all_predicted_money / all_actual_money)*100 -
100):.2f}%','%.3f'% mae_model,'%.3f'%rmse_model,'%.3f'% weighted_mape,'\n')

preds_proba = model.predict_proba(X_test)
lr_auc = roc_auc_score(y_test.values, preds_proba[:,1])

tmp = test[['gross', 'reg_dt', 'recur_dt', 'is_success',  'channel_id']]
tmp['actual_money'] = tmp.gross * tmp.is_success
tmp['model_probability'] = preds_proba[:,1]
tmp['predicted_money'] = tmp.gross * tmp.model_probability

all_predicted_money = tmp.predicted_money.sum()

res = tmp.groupby(['reg_dt',
'channel_id'])['actual_money','predicted_money'].sum().reset_index()
```

```
mae_model = mae(res.actual_money, res.predicted_money)
rmse_model = np.sqrt(mse(res.actual_money, res.predicted_money))

r = res.drop(res[res.actual_money == 0].index)
weighted_mape = sum(abs(r.actual_money - r.predicted_money)) /
sum(r.actual_money) * 100



print('%.2f'%lr_auc,f'{((all_predicted_money / all_actual_money)*100 -
100):.1f}%','%.2f'% mae_model,'%.2f'%rmse_model,'%.2f'% weighted_mape)

print(classification_report(y_test.values, model.predict(X_test)))
print('confusion_matrix:')
tn, fp, fn, tp = confusion_matrix(y_test.values, model.predict(X_test)).ravel()
print(f"'
--{tp}--{fp}
--{fn}--{tn}"')

preds_proba = model.predict_proba(X_test)
lr_auc = roc_auc_score(y_test.values, preds_proba[:,1])
print('\nROC AUC=%.3f\n' % (lr_auc))
lr_fpr, lr_tpr, _ = roc_curve(y_test.values, preds_proba[:,1])
pyplot.plot(lr_fpr, lr_tpr, marker='.', label='Logistic')
pyplot.xlabel('False Positive Rate')
pyplot.ylabel('True Positive Rate')
pyplot.legend()
```

```
pyplot.show()

tmp = test[['gross', 'reg_dt', 'recur_dt', 'is_success',  'channel_id']]
tmp['actual_money'] = tmp.gross * tmp.is_success
tmp['model_probability'] =
model.predict_proba(test.drop(columns=drop_columns))[:,1]
tmp['predicted_money'] = tmp.gross * tmp.model_probability

all_actual_money = tmp.actual_money.sum()
all_predicted_money = tmp.predicted_money.sum()
print(f'Actual money: {all_actual_money:.2f}\nPredicted Money:
{all_predicted_money:.2f}\n')

res = tmp.groupby(['reg_dt',
'channel_id'])['actual_money','predicted_money'].sum().reset_index()
mae_model = mae(res.actual_money, res.predicted_money)
rmse_model = np.sqrt(mse(res.actual_money, res.predicted_money))
print(f'MAE: {mae_model:.2f}\nRMSE: {rmse_model:.2f}\n')

(res.actual_money - res.predicted_money).plot.hist(xlim = (-700,700), alpha=0.5,
figsize = (13,7), bins = 200)
plt.axvline(x=0)

r = res.drop(res[res.actual_money == 0].index)
weighted_mape = sum(abs(r.actual_money - r.predicted_money)) /
sum(r.actual_money) * 100
```

```
print(f'w-MAPE: {weighted_mape:.2f}%')


preds_proba = model.predict_proba(X_test)
lr_auc = roc_auc_score(y_test.values, preds_proba[:,1])
print('ROC AUC=%.3f' % (lr_auc))
lr_fpr, lr_tpr, _ = roc_curve(y_test.values, preds_proba[:,1])
pyplot.plot(lr_fpr, lr_tpr, marker='.', label='Logistic')
pyplot.xlabel('False Positive Rate')
pyplot.ylabel('True Positive Rate')
pyplot.legend()
pyplot.show()
# ROC AUC=0.829

main_channels = [1645, 1664, 1730, 1034, 1684, 1700, 1549, 1489, 1513, 250,
1692, 1510, 1791]
tmp = test[['gross', 'reg_dt', 'recur_dt', 'is_success',
'channel_id']]#.query(f"channel_id in {main_channels}")


tmp['actual_money'] = tmp.gross * tmp.is_success
tmp['model_probability'] =
model.predict_proba(test.drop(columns=drop_columns))[:,1]
tmp['predicted_money'] = tmp.gross * tmp.model_probability
all_actual_money = tmp.actual_money.sum()
all_predicted_money = tmp.predicted_money.sum()
```

```
res = tmp.groupby(['reg_dt',
'channel_id'])['actual_money','predicted_money'].sum().reset_index()
# res.drop(res[res.actual_money == 0].index, inplace = True)
print(f'Actual money: {all_actual_money}\nPredicted Money:
{all_predicted_money:.2f}')

mae_model = mae(res.actual_money, res.predicted_money)
rmse_model = np.sqrt(mse(res.actual_money, res.predicted_money))

mae_model, rmse_model
print(f'MAE: {mae_model}\nRMSE: {rmse_model}\n')

(res.actual_money - res.predicted_money).plot.hist(xlim = (-700,700), alpha=0.5,
figsize = (13,7), bins = 200)
plt.axvline(x=0)

def mean_absolute_percentage_error(y_true, y_pred):
    y_true, y_pred = np.array(y_true), np.array(y_pred)
    return np.mean(np.abs((y_true - y_pred) / y_true)) * 100

r = res.drop(res[res.actual_money == 0].index)
mape = mean_absolute_percentage_error(r.actual_money.values,
r.predicted_money.values)
weighted_mape = sum(abs(r.actual_money - r.predicted_money)) /
sum(r.actual_money) * 100
print(f'MAPE: {mape:.2f}%\nw-MAPE: {weighted_mape:.2f}%')
```

```
model.get_feature_importance(prettified=True).iloc[14:, 0].values

r = tmp.groupby('recur_dt')[['actual_money', 'predicted_money']].sum()

sum((r.actual_money - r.predicted_money) > 0)

res.actual_money.plot.hist(xlim = (0,1000), alpha=0.5, figsize = (20,10), bins = 200)

delimeter_date = '2019-12-31'

# [(df.tariff == 31) | ((df.tariff == 30) & (df.type_id == 3))]

train = df[(pd.to_datetime(df.recur_dt) < pd.to_datetime(delimeter_date)) & (df.reg_dt < pd.to_datetime(delimeter_date)) & ((df.tariff == 31) | ((df.tariff == 30) & (df.type_id == 3)))]

test = df[(pd.to_datetime(df.recur_dt) >= pd.to_datetime(delimeter_date)) & (df.reg_dt >= pd.to_datetime(delimeter_date)) & ((df.tariff == 31) | ((df.tariff == 30) & (df.type_id == 3)))]


X_train = train.drop(columns=['is_success']).drop(columns=['reg_dt', 'description', 'channel_id', 'country_code', 'recur_dt', 'parent_order_id', 'net'])

y_train = train.is_success


X_test = test.drop(columns=['is_success','reg_dt', 'description', 'channel_id', 'country_code', 'recur_dt', 'parent_order_id', 'net'])

y_test = test.is_success



print(f'train: \n{y_train.value_counts()}\n')

k =  y_train.value_counts().iloc[1] / y_train.value_counts().iloc[0]
```

```
print('%.3f' % (y_train.value_counts().iloc[1] / y_train.value_counts().iloc[0]))

print(f'\ntest: \n{y_test.value_counts()}')
print('\n%.3f' % (y_test.value_counts().iloc[1] / y_test.value_counts().iloc[0]), '\n')


cat_features = X_train.columns[X_train.dtypes == np.object]

train_pool = Pool(
    data=X_train,
    label=y_train,
    weight=(y_train * (k - 1) + 1).values,
    cat_features=cat_features

)

test_pool = Pool(
    data=X_test,
    label=y_test,
    cat_features=cat_features
)


# Initialize CatBoostClassifier
model = CatBoostClassifier(learning_rate=0.03,
```

```
                    iterations=300,
                    custom_loss=['AUC', 'Accuracy', 'Recall', 'Precision'],
                    eval_metric='F1',
                    depth=6,
#                     thread_count=8,
                    subsample=0.85,
                    l2_leaf_reg=3.0,
                    rsm=0.75
                   )
# Fit model

model.fit(train_pool,eval_set=test_pool, verbose=100, plot=True)

# Get predicted probabilities for each class
preds_proba = model.predict_proba(X_test)

print(classification_report(y_test.values, model.predict(X_test)))
print('confusion_matrix:')
tn, fp, fn, tp = confusion_matrix(y_test.values, model.predict(X_test)).ravel()
print(f"'
--{tp}--{fp}
--{fn}--{tn}"')

preds_proba = model.predict_proba(X_test)
lr_auc = roc_auc_score(y_test.values, preds_proba[:,1])
```

```
print('\nROC AUC=%.3f' % (lr_auc))
lr_fpr, lr_tpr, _ = roc_curve(y_test.values, preds_proba[:,1])
pyplot.plot(lr_fpr, lr_tpr, marker='.', label='Logistic')
pyplot.xlabel('False Positive Rate')
pyplot.ylabel('True Positive Rate')
pyplot.legend()
pyplot.show()


main_channels = [1645, 1664, 1730, 1034, 1684, 1700, 1549, 1489, 1513, 250, 1692, 1510, 1791]
tmp = test[['gross', 'reg_dt', 'is_success',  'channel_id']].query(f"channel_id not in {main_channels}")

tmp['actual_money'] = tmp.gross * tmp.is_success
tmp['model_probability'] = model.predict_proba(test.query(f"channel_id not in {main_channels}").drop(columns=['is_success','reg_dt', 'description', 'channel_id', 'country_code', 'recur_dt', 'parent_order_id', 'net']))[:,1]
tmp['predicted_money'] = tmp.gross * tmp.model_probability
all_actual_money = tmp.actual_money.sum()
all_predicted_money = tmp.predicted_money.sum()

res = tmp.groupby(['reg_dt', 'channel_id'])['actual_money','predicted_money'].sum().reset_index()
print(f'Actual money: {all_actual_money}\nPredicted Money: {all_predicted_money:.2f}\n')
```

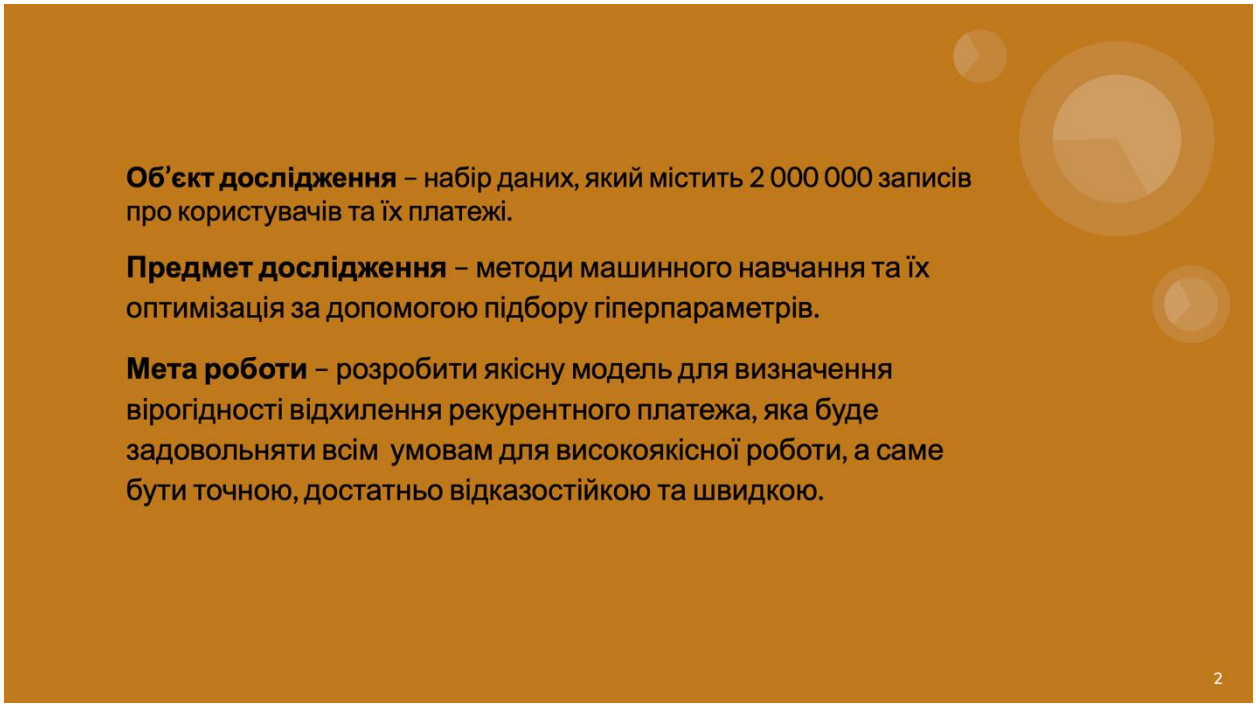
```
mae_model = mae(res.actual_money, res.predicted_money)
rmse_model = np.sqrt(mse(res.actual_money, res.predicted_money))
print(f'MAE: {mae_model}\nRMSE: {rmse_model}\n')

(res.actual_money - res.predicted_money).plot.hist(xlim = (-200,200), alpha=0.5,
figsize = (10,10), bins = 130)
plt.axvline(x=0)

mape = mean_absolute_percentage_error(r.actual_money.values,
r.predicted_money.values)
weighted_mape = sum(abs(r.actual_money - r.predicted_money)) /
sum(r.actual_money) * 100
print(f'MAPE: {mape:.2f}%\nw-MAPE: {weighted_mape:.2f}%')

model.get_feature_importance(prettified=True)
```

# ДОДАТОК Б ІЛЮСТРАТИВНИЙ МАТЕРІАЛ

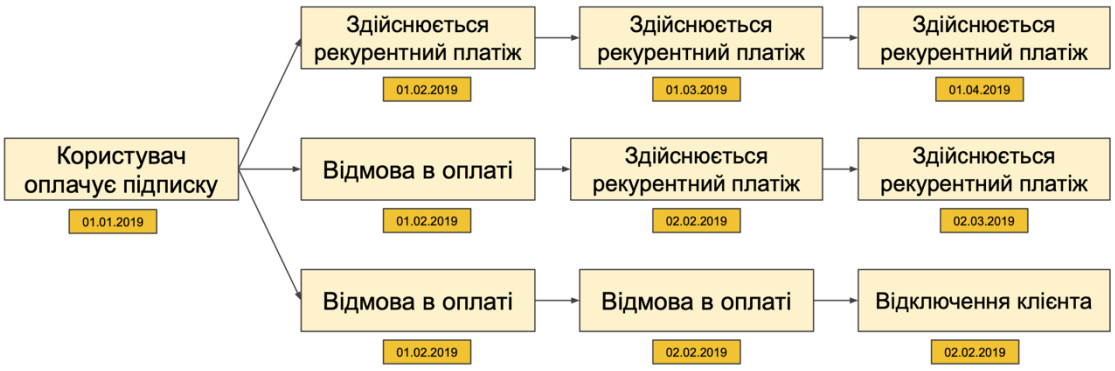
# Огляд предметної області

Підписна бізнес-модель - це бізнес-модель, в якій замовник або користувач повинен регулярно платити ціну, що повторюється, за доступ до товару або послуги.

Рекурентні платежі - це особливий вид платежів, які регулярно виконують списання коштів з банківської картки користувача сервісу без необхідності кожного разу вводити заново реквізити картки і без особистої участі платника для здійснення операції.



# Основні варіанти розвитку подій



# Причини відмови у здійсненні платежу

1. Недостатньо коштів на картці
2. Відмова від банку
3. Закінчився строк придатності картки
4. Картка викрадена
5. Картка загублена
6. Антифрод система відмінила платіж
7. Технічний збій

5

# Бустинг

Бустинг - це підхід до побудови композицій, в рамках якого:

- Базові алгоритми будуються послідовно, один за іншим.
- Кожний наступний алгоритм будується таким чином, щоб виправляти помилки вже побудованої композиції.

Завдяки тому, що побудова композицій в бустингу є послідовною, достатньо використовувати прості базові алгоритми, наприклад, неглибокі дерева.

6

## Реалізація бустингу

Нехай є функція помилки: $$MSE(a, X) = \frac{1}{\ell}\sum_{i=1}^{\ell}\left(a(x_i) - y_i\right)^2.$$

Для початку потрібно побудувати перший базовий алгоритм: $$b_1(x) = \operatorname{argmin}_b \frac{1}{\ell}\sum_{i=1}^{\ell}\left(b(x_i) - y_i\right)^2.$$

7

## Реалізація бустингу

Другий алгоритм повинен бути навчений таким чином, щоб композиція першого і другого алгоритмів мала найменш можливу помилку на навчальній вибірці:

$$b_2(x) = \operatorname{argmin}_b \frac{1}{\ell}\sum_{i=1}^{\ell}\left(b_1(x_i) + b(x_i) - y_i\right)^2 = \operatorname{argmin}_b \frac{1}{\ell}\sum_{i=1}^{\ell}\left(b(x_i) - (y_i - b_1(x_i))\right)^2.$$

8

# Реалізація бустингу

Іншими словами, другий алгоритм (b2) покращує якість роботи першого алгоритму(b1). Продовжуючи за аналогією, на N кроці черговий алгоритм bn буде визначатися таким чином:

$$b_N(x) = \text{argmin}_b \frac{1}{\ell} \sum_{i=1}^{\ell} \left( b(x_i) - \left( y_i - \sum_{n=1}^{N-1} b_n(x_i) \right) \right)^2$$

Процес триває до тих пір, поки помилка композиції всіх алгоритмів не буде задовільною.

9

# Градієнтний бустинг

Метод градієнтного бустингу вважається одним з найкращих способів спрямованої побудови композиції на сьогоднішній день. У градієнтному бустингу композиція є сумою всіх базових алгоритмів:

$$a_N(x) = \sum_{n=1}^{N} b_n(x)$$

10

# Алгоритм побудови

**Ініціалізація:** ініціалізація композиції, тобто побудова простого алгоритму b0.

$$a_0(x) = b_0(x),$$

11

# Алгоритм побудови

**Крок ітерації:** (а) обчислюється вектор зсуву $s = -\nabla F = \begin{pmatrix} -L'_z(y_1, a_{n-1}(x_1)), \\ \dots \\ -L'_z(y_\ell, a_{n-1}(x_\ell)) \end{pmatrix}.$

(б) будується алгоритм $b_n(x) = \operatorname{argmin}_b \frac{1}{\ell} \sum_{i=1}^{\ell} (b(x_i) - s_i)^2$

Параметри якого підбираються таким чином, щоб його значення на навчальній вибірці були якомога ближче до обчисленого вектору оптимального зсуву s.

12

## Алгоритм побудови

(в) Алгоритм додається в композицію $$a_n(x) = \sum_{m=1}^{n} b_m(x)$$

Якщо не виконано критерій зупинки, то виконати ще один крок ітерації. Якщо критерій зупинки виконаний, зупинити ітераційний процес.

13

# Результати роботи

14

# Початкова модель

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.87 | 0.98 | 0.92 | 241309 |
| 1 | 0.59 | 0.16 | 0.26 | 42314 |
| accuracy | | | 0.86 | 283623 |
| macro avg | 0.73 | 0.57 | 0.59 | 283623 |
| weighted avg | 0.83 | 0.86 | 0.82 | 283623 |

confusion_matrix:

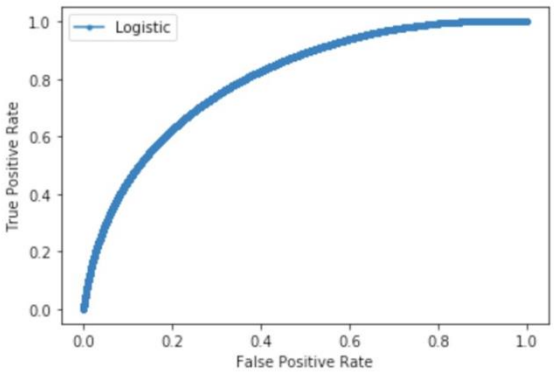
--6895--4850
--35419--236459

15

# Початкова модель

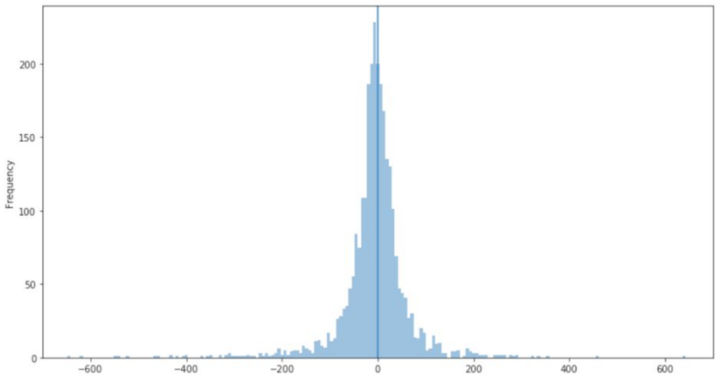
DEV = 7.3249%

ROC AUC=0.801




16

## Початкова модель




Розподіл похибки

17

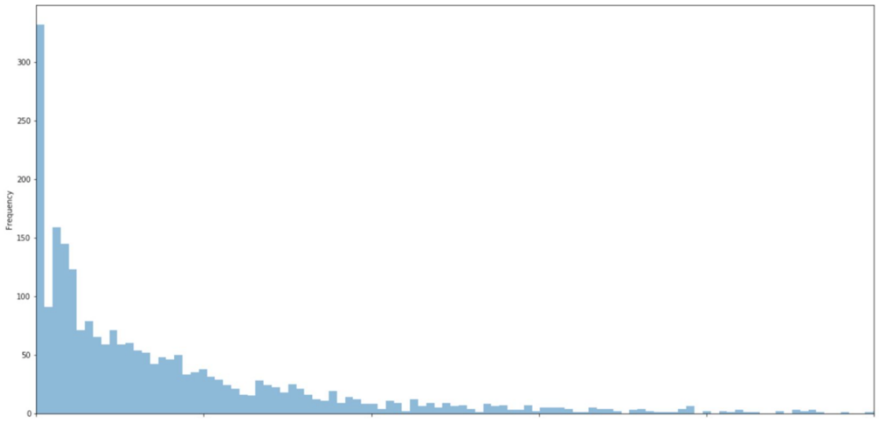
## Початкова модель

MAE: 54.34562234
RMSE: 109.208438

MAPE: 45.34%
w–MAPE: 21.46%

18

# Розподіл прибутків




19

# Вага параметрів

```
model.get_feature_importance(prettified=True)
```

| | Feature Id | Importances |
|---|---|---|
| 0 | previous_response | 42.776385 |
| 1 | description | 21.669330 |
| 2 | success_orders_before_rec | 5.958408 |
| 3 | card_issuer | 3.984794 |
| 4 | net | 3.611474 |
| 5 | country_group | 3.145448 |
| 6 | retry_number | 3.104263 |
| 7 | gross | 2.738542 |
| 8 | days_after_reg | 2.594875 |
| 9 | sessions_in_5_days_before_rec | 2.134494 |
| 10 | card_type | 1.864427 |
| 11 | payment_type | 1.338369 |
| 12 | age | 1.038223 |
| 13 | card_brand | 1.017631 |
| 14 | cpa_subchannel | 0.486234 |
| 15 | gender | 0.485108 |
| 16 | context_id | 0.440741 |
| 17 | descriptor | 0.428107 |
| 18 | discount | 0.274982 |
| 19 | pay_app | 0.243623 |
| 20 | type_trafic | 0.178240 |
| 21 | merchant_id | 0.174210 |
| 22 | tariff | 0.143628 |
| 23 | country_code | 0.134638 |
| 24 | app | 0.033827 |

20

# Таблиця результатів

| Параметри моделі | | | | Метрики | | | | |
|---|---|---|---|---|---|---|---|---|
| reg | subsample | rsm | depth | auc | Dev | MAE | RMSE | w-MAPE |
| 4 | 0.7 | 0.85 | 4 | 0.79381 | 6.77% | 51.587 | 95.123 | 17.219 |
| 4 | 0.7 | 0.85 | 12 | 0.80089 | 5.69% | 48.788 | 88.555 | 16.259 |
| 4 | 0.7 | 1 | 4 | 0.79356 | 6.72% | 51.681 | 95.587 | 17.258 |
| 4 | 0.7 | 1 | 12 | 0.80078 | 5.60% | 48.579 | 87.512 | 16.170 |
| 4 | 0.85 | 0.85 | 4 | **0.79355** | 5.80% | 50.634 | 92.396 | 16.891 |
| 4 | 0.85 | 0.85 | 12 | 0.80070 | 5.93% | 49.118 | 88.950 | 16.357 |
| 4 | 0.85 | 1 | 4 | 0.79391 | 6.37% | 51.173 | 94.089 | 17.083 |
| 4 | 0.85 | 1 | 12 | 0.80106 | **5.33%** | **48.221** | **86.632** | **16.050** |
| 7 | 0.7 | 0.85 | 4 | **0.79320** | 7.09% | 52.091 | 96.202 | 17.391 |
| 7 | 0.7 | 0.85 | 12 | 0.80092 | **5.52%** | **48.472** | **87.253** | **16.132** |
| 7 | 0.7 | 1 | 4 | 0.79383 | 6.35% | 51.064 | 93.903 | 17.047 |
| 7 | 0.7 | 1 | 12 | 0.80019 | 5.78% | 48.914 | 87.847 | 16.265 |
| 7 | 0.85 | 0.85 | 4 | 0.79379 | 6.15% | 50.837 | 93.037 | 16.959 |
| 7 | 0.85 | 0.85 | 12 | 0.80040 | 5.99% | 49.078 | 89.260 | 16.343 |
| 7 | 0.85 | 1 | 4 | **0.79338** | 6.60% | 51.539 | 95.116 | 17.210 |
| 7 | 0.85 | 1 | 12 | 0.80069 | **5.39%** | **48.378** | **86.627** | **16.091** |

21

# Прийняття рішень на основі прогнозу

| | Витрати на рекламу | Заохочено користувачів | Дійсних підписок | Прибуток (максимальний) | Рішення | Прибуток (вірогідний) | Рішення |
|---|---|---|---|---|---|---|---|
| Facebook ad #1 | 7000 $ | 26 000 | 2 050 | 10 250 | Масштабування | 8 500 | Масштабування |
| Facebook ad #2 | 7000 $ | 28 000 | 2 150 | 10 750 | Масштабування | 6 000 | Оптимізація / зупинка |

Рішення
без використання моделі

Рішення
із використанням моделі

22

# Висновки


- Запропоновано та реалізовано градієнтний бустинг над деревами рішень
- Розроблений алгоритм отримав значення метрики w-MAPE **16.091%,** що перевищує результат початкової моделі
- **Даний алгоритм реалізовано в компанії Genesis**

23

# Дякую за увагу!

24